NADC-77163-30

A COMPUTATIONAL MODEL FOR THREE-DIMENSIONAL INCOMPRESSIBLE WALL JETS WITH LARGE CROSS FLOW

by

W.D. MURPHY, V. SHANKAR, and N. MALMUTH

SCIENCE CENTER
ROCKWELL INTERNATIONAL
THOUSAND OAKS, CA 91360

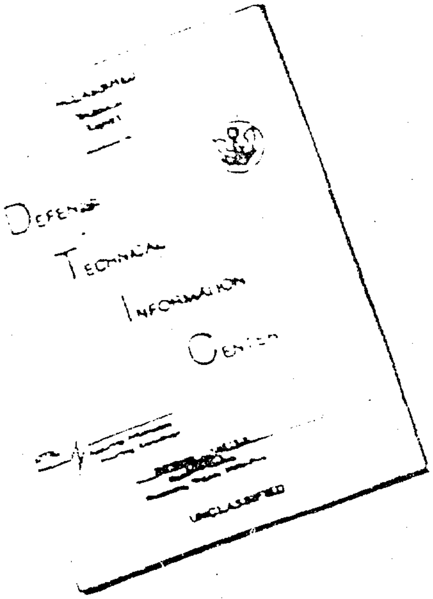
SEPTEMBER 1978

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

PREPARED FOR

NAVAL AIR DEVELOPMENT CENTER
WARMINSTER, PENNSYLVANIA 18974

DISCLAIMER NOTICE



THIS DOCUMENT IS BEST QUALITY AVAILABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

REPRODUCED FROM
BEST AVAILABLE COPY

(11) REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER
(14) NADC-77163-30

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
(6) A COMPUTATIONAL MODEL FOR THREE-DIMENSIONAL INCOMPRESSIBLE WALL JETS WITH LARGE CROSS FLOW.

5. TYPE OF REPORT & PERIOD COVERED
Final Report, 19 Sep 77 — through 18 Sep 78

6. PERFORMING ORG. REPORT NUMBER
(14) SC5129.5FR

7. AUTHOR(s)
(10) W.D. Murphy, V. Shankar ~~and~~ N. Malmuth

8. CONTRACT OR GRANT NUMBER(s)
(15) N62269-77-C-0412

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Science Center
Rockwell International
Thousand Oaks, CA 91360

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
(16) P.E. — 61153N
~~PROJ. — RR02302~~
(17) ~~T.A. — RR0230221~~
W.U. — NN701BB

11. CONTROLLING OFFICE NAME AND ADDRESS
(11) Naval Air Development Center
Code 3014
Warminster, PA 18974

12. REPORT DATE
September 1978

13. NUMBER OF PAGES
146

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)
(12) 149p.

15. SECURITY CLASS. (of this report)
UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)
Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Incompressible Flow
Viscous Flow
Jets
Partial Differential Equations
Numerical Analysis
Ejectors

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The flow field of three dimensional incompressible wall jets prototypic of thrust augmenting ejectors with large cross flow is solved using a very efficient centered-Euler scheme in an orthogonal curvilinear coordinate system. The computational model treats initial conditions with arbitrary velocity profiles at or downstream of the jet exit. An averaging approach is employed for the first few marching steps to overcome spurious numerical oscillations associated with arbitrary initial profiles. Laminar as well as turbulent wall jets are

389 949 LB

Block 20 Continued:

simulated. Turbulence is introduced using a two layer mixing length model appropriate to curved three-dimensional wall jets. Typical results quantifying jet spreading, jet growth, nominal separation and jet shrink effects due to cross flow are presented.

A class of cases was investigated roughly possessing initial flow angularity and adverse pressure gradients prototypic of those on the blown surfaces of typical propulsive lift systems such as the Navy/Rockwell XFV-12A thrust augmented wing. Results obtained from the computational model indicate that if the initial total velocity is kept fixed, then the introduction of the cross flow enhances the freestream decay rate of the peak of the velocity component in the freestream direction. In addition, the entrainment quantity and its rate decrease with increased cross flow. The three-dimensional phenomena not only influence the effect of taper on the boundary layer control characteristics of a Coanda flap, but also indicate a "jet shrink" which could be a mechanism promoting end-wall separation. To our knowledge, our model is the first to quantify such trends. Both should be considered in the design of any propulsive lift system. Finally, the effect on the prescribed external adverse pressure gradient in the presence and absence of cross flow has also been examined. From the limited results, the spanwise separation line moves progressively further upstream with increasing cross flow.

NADC-77163-30

A COMPUTATIONAL MODEL FOR THREE-DIMENSIONAL INCOMPRESSIBLE WALL JETS WITH LARGE CROSS FLOW

by

W.D. MURPHY, V. SHANKAR, and N. MALMUTH

SCIENCE CENTER
ROCKWELL INTERNATIONAL
THOUSAND OAKS, CA 91360

SEPTEMBER 1978

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

PREPARED FOR

NAVAL AIR DEVELOPMENT CENTER
WARMINSTER, PENNSYLVANIA 18974

FOREWORD

This document describes computational studies of three-dimensional incompressible turbulent wall jets with large cross flow. The effort was performed during the period September 19, 1977 to September 18, 1978 and was sponsored by the Naval Air Development Center under Contract N62269-77-C-0412.

The technical monitor for this study was Dr. K.A. Green.

ABSTRACT

The flow field of three dimensional incompressible wall jets prototypic of thrust augmenting ejectors with large cross flow is solved using a very efficient centered-Euler scheme in an orthogonal curvilinear coordinate system. The computational model treats initial conditions with arbitrary velocity profiles at or downstream of the jet exit. An averaging approach is employed for the first few marching steps to overcome spurious numerical oscillations associated with arbitrary initial profiles. Laminar as well as turbulent wall jets are simulated. Turbulence is introduced using a two layer mixing length model appropriate to curved three-dimensional wall jets. Typical results quantifying jet spreading, jet growth, nominal separation and jet shrink effects due to cross flow are presented.

A class of cases was investigated roughly possessing initial flow angularity and adverse pressure gradients prototypic of those on the blown surfaces of typical propulsive lift systems such as the Navy/Rockwell XFV-12A thrust augmented wing. Results obtained from the computational model indicate that if the initial total velocity is kept fixed, then the introduction of the cross flow enhances the freestream decay rate of the peak of the velocity component in the freestream direction. In addition, the entrainment quantity and its rate decrease with increased cross flow. The three-dimensional phenomena not only influence the effect of taper on the boundary layer control characteristics of a Coanda flap, but also indicate a "jet shrink" which could be a mechanism promoting end-wall separation. To our knowledge, our model is the first to quantify such trends. Both should be considered in the design of any propulsive lift system. Finally, the effect on the prescribed external adverse pressure gradient in the presence and absence of cross flow has also been examined. From the limited results, the spanwise separation line moves progressively further upstream with increasing cross flow.

CONTENTS

Page

FOREWORD v

ABSTRACT vii

CONTENTS ix

ILLUSTRATIONS xi

NOMENCLATURE xiii

PART I — THEORETICAL ANALYSIS AND RESULTS

1.0 Introduction I-1

2.0 Problem Description and Formulation I-4

3.0 Eddy-Viscosity Model I-8

4.0 Finite Difference Equations I-13

5.0 Results I-19

6.0 Conclusions I-30

7.0 References I-32

PART II — USERS MANUAL

1. Deck Setup II-1

2. Estimate of Running Time II-2

3. Type and Configuration of Computer Used in
Program Development II-2

4. Name and Level of Programming Language Used in Program II-2

A. Input-Output Information II-2

Glossary of Input Parameters II-2

Input II-5

CONTENTS (Continued)

Page

Plots II-6

Example II-7

Output II-7

Example II-9

B. Subroutine Description II-11

BC II-11
BETASV II-11
BLOCK1 . . . . II-13
BLOCK2 . . . . II-13
BOX II-14
LUSOLV II-14
MAIN II-14
NETRUN II-15
NEWTON II-15
OUTPT II-16
PREP II-17
PREPB II-17
PREPG II-18
PREPG2 . . . . II-18
PREPP* II-19
PRMESH*. II-20
PROFLE*. II-20
RHSF II-21
RHSF2D II-21
TRUN II-22
USOLVE II-22
VANDET II-22
VELMAX II-23
WALJET II-23
XMESH* II-24
YMESH* II-24
ZMESH* II-24

APPENDIX — PROGRAM LISTING A-1

ILLUSTRATIONS

Figure	Title	Page
1	XFV-12A Prototypic Augmenter Configuration	I-5
2	Upper Surface Blowing Boundary Layer Control-Supercirculation Wing	I-5
3	Physical System and Flow Schematic	I-7
4	Two-Layer Eddy Viscosity Model	I-10
5	Initial and Boundary Conditions	I-18
6	Effect of Cross Flow on Jet Growth, $\frac{u_{max}}{u_e} = 2 - \|.5-z\|$, $u_e = u_o x^{-1/2} \cos\theta$, $\theta = \frac{\pi}{2} z(z_{tip} - z)$, Standard Initial Profile (See Fig. 5), $z = z_1 = z_{tip}/2$	I-21
7	Upper Surface Blown Thrust Augmented Wing (TAW) . . .	I-22
8	Effect of Cross Flow on Nominal Entrainment, Q with $u_e = u_o x^{-1/2} \cos\theta$, $Q = \int_0^\infty \int_0^{z_{tip}} u dy dz$, $\theta = C\frac{\pi}{2} z(z_{tip} - z)$, $\frac{u_{max}}{u_e} = 2 - \|.5-z\|$, (Standard Initial Profile) $z = z_1 = z_{tip}/2$	I-24
9	Effect of Cross Flow on Reduced Shear Stress, $u_e = u_o x^{-1/2} \cos\theta$, $\theta = C\frac{\pi}{2} z(z_{tip} - z)$, Standard Initial Profile, $(z = z_1 = z_{tip}/2)$, $\frac{u_{max}}{u_e} = 2 - \|.5-z\|$	I-25

ILLUSTRATIONS (Continued)

Figure | Title | Page

10 Effect of Cross Flow on Jet Spreading, $u_e = u_o x^{-1/2} \cos\theta$, $\theta = C\frac{\pi}{2} z(z_{tip} - z)$, Standard Initial Profile at Midspan ($z = z_{tip}/2$), $\frac{u_{max}}{u_e} = 2 - |.5-z|$ I-27

11 Effect of Cross Flow on the Locus of Nominal Separation, $u_e = u_o x^{-1/2} \cos\theta$, $w_e = u_o x^{-1/2} \sin\theta$, $\theta = C\frac{\pi}{2} z(z_{tip} - z)$, $\frac{u_{max}}{u_e} = 2 - |.5-z|$ I-28

12 Cross Flow Effect on Jet "Shrink" and "End Wall Pullaway," $u_e = u_o x^{-1/2} \cos\theta$, $w_e = u_o x^{-1/2} \sin\theta$, $\frac{u_{max}}{u_e} = 2 - |.5-z|$ I-29

13 Cross Flow Effect on Jet "Shrink" with Streamwise Increase in Cross Flow, $u_e = u_o x^{-1/2} \cos\theta$, $w_e = u_o x^{1/2} \sin\theta$, $\frac{u_{max}}{u_e} = 2 - |0.5 - z|$, $\theta = \frac{c\pi}{2} z(z_{tip} - z)$ I-31

NOMENCLATURE

b	Defined in Eq. (14a)
f,g	Reduced vector potentials (Eqs. (8a) and (8b))
h_1,h_2,h_3	Metric coefficients
K_1,K_2	Geodesic curvatures
$P_i,(i=1,\ldots,10)$	Constants defined on page I-12
M,N,Q,R	Constants defined on page I-12
Q	Entrainment quantity
u,v,w	x,y,z components of velocity (see Fig. 3)
$\overline{u'v'}, \overline{v'w'}$	Reynolds stress correlations
S_1	Arc length along x coordinate
S	Area of integration
(x,y,z)	Orthogonal curvilinear coordinates parallel and perpendicular to wall (see Fig. 3)
y^*	Height of first turbulent layer in two layer eddy viscosity model
y_1	Constant appearing in eddy viscosity model, page I-9
ε	Eddy viscosity (page I-9)
$\bar{\varepsilon}$	Reduced eddy viscosity (page I-9)
ε^+	Reduced oddy viscosity defined in Eq. (14a)
κ_1,κ_2	Radius of curvature of z=constant and x=constant lines on wall jet surface
ϕ	Component of vector potential (page I-8), velocity potential (page I-20)
ψ	Component of vector potential (page I-8)

NOMENCLATURE (Continued)

σ	Source strength (page I-20)
ν	Kinematic viscosity
τ	Shear stress = $\rho(\varepsilon+\nu)\sqrt{u_y^2 + w_y^2}$
θ	Initial jet flow angle parameter, Fig. 8
ξ,ζ	Dummy variables, page I-20

Subscripts

e	External flow
i,j	Nodal locations for difference operators (Note also that subscript notation is used in what follows to denote partial differentiation.)

Superscript

n	Time level and nodal location for difference operators

PART I — THEORETICAL ANALYSIS AND RESULTS

1.0 Introduction

Modern naval aircraft can reduce strike force vulnerability through greater dispersal. A way of achieving this dispersal would be the development and deployment of vertical lift-off and landing aircraft operating from ships that are significantly smaller than the currently operating carriers. One propulsion concept designed to achieve a vertical and short take-off and landing (V/STOL) aircraft involves the use of thrust augmenting ejectors to amplify the thrust of the engines in the vertical mode. This technique is utilized in the current XFV-12A aircraft. To achieve adequate accelerations for typical payloads, a high augmentation ratio (ϕ) is required. Various augmenter designs have been proposed to achieve these high ϕ values as well as integrate well with the aircraft structure. In the XFV-12A, an ejector system composed of a centerbody and two Coanda wall jets is currently under development. A central feature of the flow fields produced by this device is three dimensionality. This has been particularly evident in subscale flow visualization on the Coanda surfaces. It is believed that these flow processes may be important toward ϕ maximization. The augmentation ratio depends on the momentum flux and the dynamic head losses downstream of the Coandas and the centerbody nozzle, and in particular, in the diffusor section of the augmenter. Maintenance of a high momentum and minimization of extraneous motions, while accelerating as quickly as possible the secondary flows using the primary jets, is of paramount importance. An example of an extraneous motion is associated with certain vortex structures which have been observed in augmenter wing configurations. It has been postulated that these motions and general flow turning are induced by the three-dimensional features of the configuration. If the wing flap on which the Coandas are mounted has a tapered diffusor, swept back trailing edge, and small aspect ratio planform, considerable cross flow in the secondary can arise. Under these circumstances, this turning is further enhanced by spanwise pressure gradients arising along the tapered centerbody and Coanda slots. These factors can also lead to considerable flow nonuniformities both in the spanwise and chordwise directions. To compound the situation, flow surveys for three dimensional research configurations of this type have indicated regions of separation near the endwalls. The amount and nature of blowing in these regions to maintain attached flow and create the highest possible momentum flux across some downstream control surface needs quantification.

From the foregoing considerations, it is obvious that an analysis which assumes mean flow quantities such as that employed in the usual one-dimensional derivations can be considerably in error. In fact, if it is assumed in such a treatment that the flow is turned by an angle ω uniformly along the span, the reduction in ϕ is only cos ω, which is considerably less than rough comparisons between quasi-two-dimensional configurations and actual three-dimensional arrangements would indicate. One way of understanding the foregoing relationships is through theoretical modeling which can provide a means of reducing the high cost of powered lift testing. Unfortunately, existing methodology has been limited in the past to two-dimensional flows for the analysis of wall jets and complete ejector systems. Analytical methods and computational algorithms are therefore necessary to compute three-dimensional flows typical of reality.

To shed light on typical flow patterns encountered due to the effect of taper and sweep on augmenter wings as well as upper-surface-blown configurations, a study, "Three-Dimensional Flow of a Wall Jet," was initiated by the Naval Air Development Center to investigate wall jet flows which exemplify typical features of more complex propulsive lift applications. The purpose of this study has been to apply modern computational methods to the treatment of wall jet flows with three dimensionality. In a previous phase of the effort, small cross flow wall jets were considered.[1] This report relaxes that assumption and considers large cross flows that occur in practice.

The formulation employs boundary layer equations in an orthogonal curvilinear coordinate system. If the distance from the jet exit is sufficiently large to establish complete mixing, the jet exit height is small compared to a characteristic radius of curvature, and the Reynolds number based on the exit height is large, the order of magnitude analysis given in Ref. 1 extended to three dimensions indicates that the wall jet equations are substantially the same as the three-dimensional boundary layer equations. The basic idea in both the wall jet and boundary layer approximations is the same, namely, that diffusional gradients for the vorticity in the direction normal to the surface are dominant terms in the equations. In fact, half of the wall jet has a boundary layer character due to the no-slip condition on the surface. Furthermore, to dominant order, the pressure is independent of the coordinate normal to the surface. Equilibration of centrifugal force between the streamlines with pressure gradients across them is accomplished with the second-order pressure term and is expressed in the second-order momentum equation in the normal direction. The complete mixing condition is synonymous with merger of the boundary layer from the wall and the shear layer from the jet exit. Assuming that the characteristic distance for merger D is determined by a spreading angle λ of the order of 2.86° appropriate for submerged turbulent free shear layers measured by Reichardt,[2] then the merger distance is d ctn λ where d is the slot height. For an aircraft similar to the XFV-12A, assuming that $d \sim 3$ cm and the wing chord $L \sim 300$ cm, this estimate shows that $D/L \sim 0.2$. Note, however, that this can be reduced to practically zero if the wall boundary layer of the flow emanating upstream of the nozzle exit completely fills the exit location.

For the computational model, a transformation is incorporated to stretch the coordinate normal to the flow. At streamwise planes, the resulting nonlinear partial differential equations are treated as ordinary differential equations, incorporating source terms involving partial derivatives representing the upstream history of the flow field. These are solved using a very efficient two-point boundary value finite-difference method devised by Keller and Cebeci[3-5] known as the "box method." In the code, the turbulence is introduced using a two layer mixing length model appropriate to three-dimensional wall jets.

2.0 Problem Description and Formulation

To fix the ideas regarding three-dimensional wall jet flows ı evant to propulsive lift applications, consider two prototypic configurations as indicated in Figures 1 and 2. The generic arrangement shown in Figure 1 is relevant to an augmenter ejector wing of the type used on the XFV-12A without the centerbody nozzle. In Figure 2, the shape indicated corresponds to an application involving boundary layer control and supercirculation development. For both cases, the development of the wall jet over the curved surfaces S_1 is of interest. In the augmenter of Figure 1, Coanda jets flow over the surfaces S_1 and S_2 emanating from slots T_1 and T_2 and provide a vertical lift force. In the analysis of the flow field over S_1, we suppress the influence of the surfaces S_2, A, and C. In addition, we assume that the secondary flow produced by entrainment resulting from the primary jets emanating from T_1 and T_2 is known *à priori*. In actuality, these must be computed as an integral part of the problem. For tractability, we restrict our consideration to the indicated (parabolic) formulation since it is a building block to a later analysis of the primary secondary interaction. For the configuration of Figure 2, the orientation is similar, and we neglect the primary secondary interaction features. Thus, both configurations lead to the problem of the development of a 3-D wall jet over a curved surface S_1 in which boundary conditions are specified on some interfacial layer with the external or entrained flow. It should be noted that the mixing downstream of the slots T_1 is with a flow above the slot which obeys the no slip condition at the slot trailing edge.

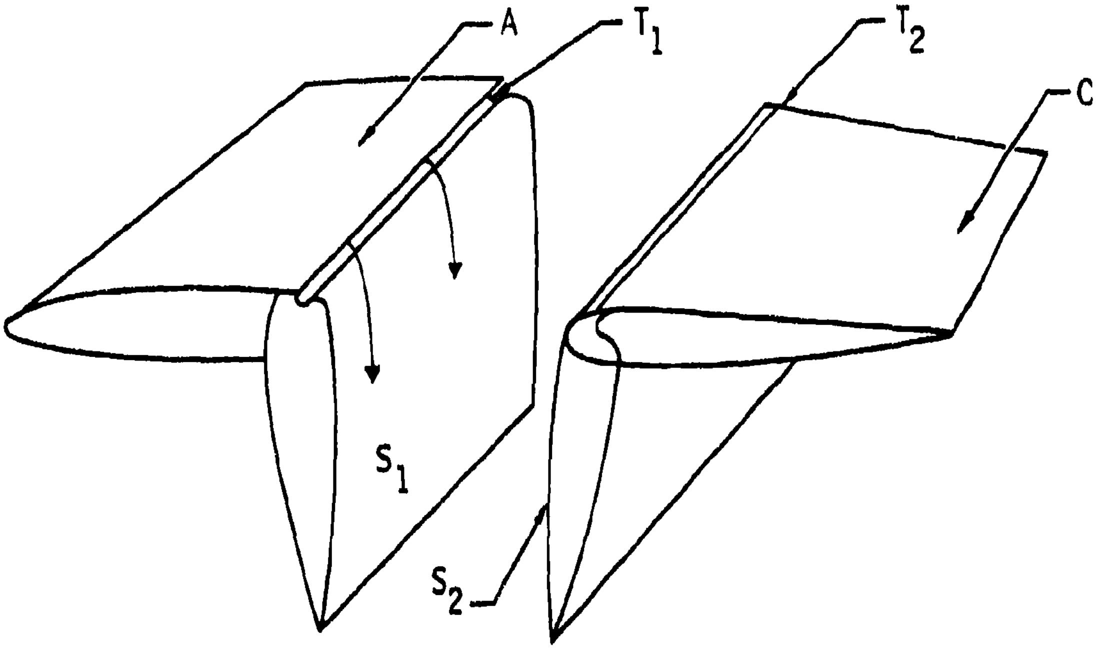


Figure 1. XFV-12A Prototypic Augmenter Configuration

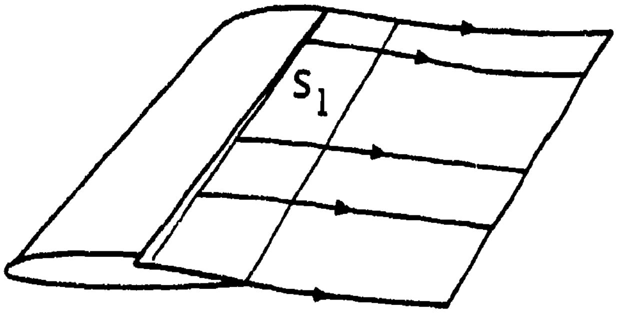


Figure 2. Upper Surface Blowing Boundary Layer Control-Supercirculation Wing

In Figure 3, the prototypic geometry for the flow over the surfaces S_1 is shown with the prescribed initial and boundary velocity distributions schematically indicated. A curvilinear coordinate system is used which consists of conveniently oriented lines in the surface and normals to them as shown in the figure. The initial distribution of velocities is specified along some line $x=x_0=$constant say, which may be the jet exit. In the wall jet/boundary layer approximation the appropriate equations describing the wall jet flow are:

Continuity

$$(\partial/\partial x)(h_2 u) + (\partial/\partial z)(h_1 w) + (\partial/\partial y)(h_1 h_2 v) = 0 . \qquad (1)$$

x-Momentum

$$\frac{u}{h_1}\frac{\partial u}{\partial x} + \frac{w}{h_2}\frac{\partial u}{\partial z} + v\frac{\partial u}{\partial y} - uwK_1 + w^2 K_2 = -\frac{1}{\rho h_1}\frac{\partial p}{\partial x} + \frac{\partial}{\partial y}\left(\nu\frac{\partial u}{\partial y} - \overline{u'v'}\right) . \qquad (2)$$

z-Momentum

$$\frac{u}{h_1}\frac{\partial w}{\partial x} + \frac{w}{h_2}\frac{\partial w}{\partial z} + v\frac{\partial w}{\partial y} - uwK_2 + u^2 K_1 = -\frac{1}{\rho h_2}\frac{\partial p}{\partial z} + \frac{\partial}{\partial y}\left(\nu\frac{\partial w}{\partial y} - \overline{w'v'}\right) . \qquad (3)$$

Here, h_1 and h_2 are metric coefficients and are functions of x and z, and the parameters K_1 and K_2 are known as the geodesic curvatures of the curves z = constant and x = constant, respectively.

The boundary conditions for Eqs. (1) through (3) for zero mass transfer at the wall and compatibility with the external flow are

$$y = 0 \qquad\qquad u,w,v = 0$$

$$y \rightarrow \infty \qquad u \rightarrow u_e(x,z) \qquad w \rightarrow w_e(x,z) . \qquad (4)$$

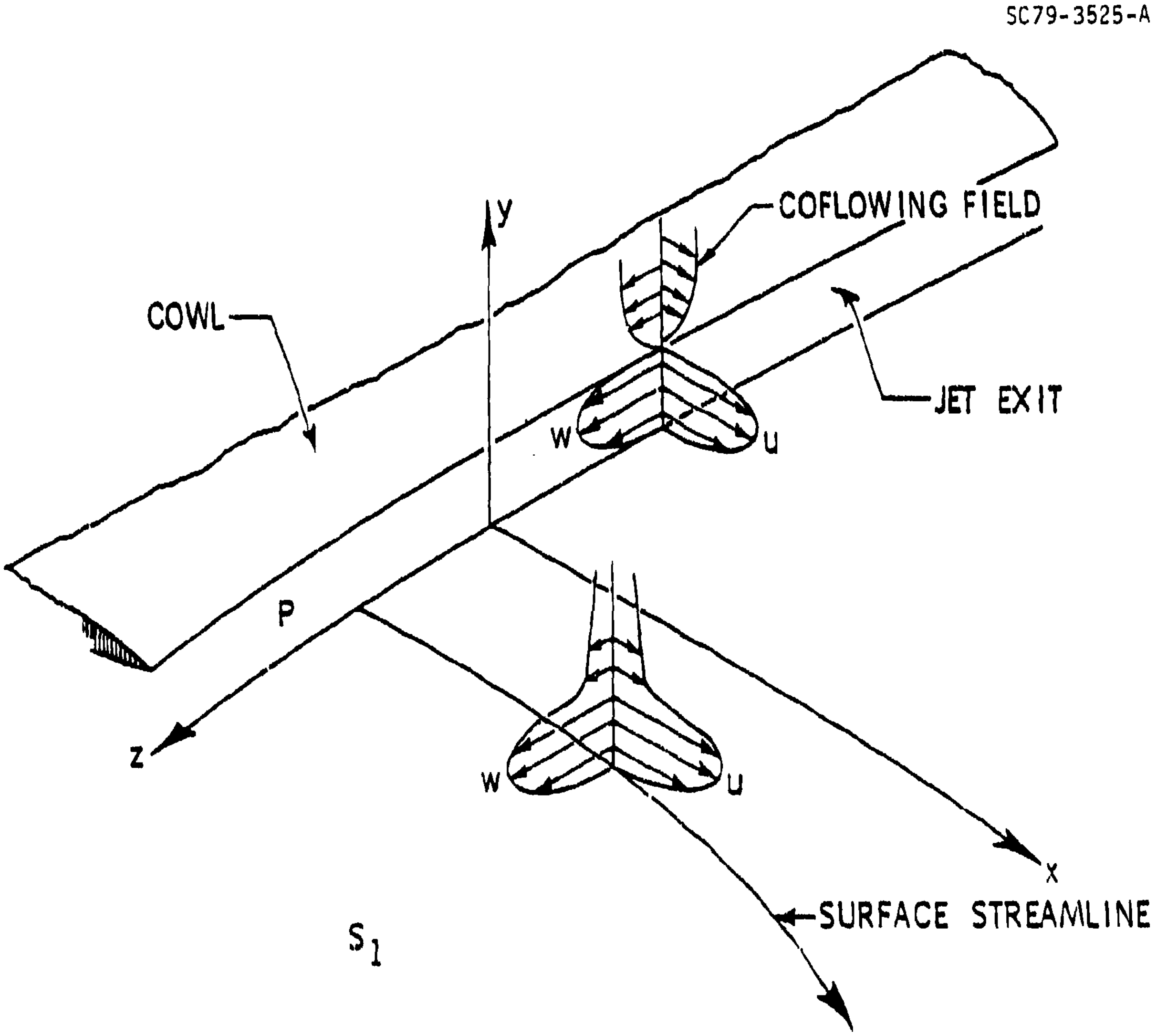


Figure 3. Physical System and Flow Schematic

As indicated earlier, the previous equations are transformed by defining

$$x = x \qquad z = z \qquad \eta = (u_e/\nu s_1)^{1/2}y \tag{5}$$

and introducing a two-component vector potential given by

$$h_2u = \frac{\partial\psi}{\partial y} \qquad h_1w = \frac{\partial\phi}{\partial y} \qquad h_1h_2v = -\left(\frac{\partial\psi}{\partial x} + \frac{\partial\phi}{\partial z}\right) . \tag{6}$$

Here s_1, which denotes the arc length along the x coordinate, is defined by

$$s_1 = \int_0^x h_1dx . \tag{7}$$

In addition, the dimensionless variables f and g related to ψ and ϕ are defined by

$$\psi = (u_e\nu s_1)^{1/2}h_2f(x,z,\eta) \tag{8a}$$

$$\phi = (u_e\nu s_1)^{1/2}h_1(w_e/u_e)g(x,z,\eta) . \tag{8b}$$

The parameters K_1 and K_2 in Eqs. (2) and (3) are defined by

$$K_1 = -\frac{1}{h_1h_2}\frac{dh_1}{dz} \qquad \text{and} \qquad K_2 = -\frac{1}{h_1h_2}\frac{dh_2}{dx} .$$

3.0 Eddy Viscosity Model

Equations (2) and (3) contain the Reynolds shear stress terms $\overline{-u'v'}$ and $\overline{-v'w'}$. These quantities must be further characterized in order to solve Eqs. (1)-(3). To achieve this closure, suitable eddy viscosity models are employed. In two dimensions, such simulations have been successfully used by Dvorak,[6] Ramaprian,[7] and Wilson and Goldstein.[8] On a heuristic basis subject to future experimental validation, extension to three-dimensional flows was effected analogous to an approach used for the same purpose in connection with boundary layer flows in Ref. 5. For the wall jet case, we assume that the curvature effects are modeled in terms of the principal curvatures in the x and z directions. With this viewpoint, the Reynolds shear stress terms are assumed to be in the form

$$\begin{aligned} \overline{-u'v'} &= \epsilon u_y = \bar{\epsilon}\left[1 - \frac{uK_2}{1+K_2 y}\,\frac{1}{u_y}\right]u_y \\ \overline{-v'w'} &= \epsilon w_y = \bar{\epsilon}\left[1 - \frac{wK_1}{1+K_1 y}\,\frac{1}{w_y}\right]w_y \ . \end{aligned} \tag{9}$$

The quantity $\bar{\epsilon}$ in Eq. (9) is the eddy viscosity of the corresponding plane flow. It is assumed to be the same in both the x and z directions and is represented by a two-layer model. The second term inside the bracket in Eq. (9) is due to curvature where K_1 and K_2 denote the radius of curvature of z=constant and x=constant lines. Similar curvature terms for two-dimensional wall jets have been used by Dvorak[6] and Wilson.[8] Referring to Figure 4, the structure of the two-layer eddy viscosity model is as follows:

<u>First Layer</u>

$$\bar{\epsilon} = (0.435\ y)^2\ \sqrt{u_y^2 + w_y^2} \qquad 0 < y < y^*$$

<u>Second Layer</u>

$$\bar{\epsilon} = (0.125\ y_1)^2\ \sqrt{u_y^2 + w_y^2} \qquad y > y^* \tag{10}$$

where at $y = y_1$

$$\frac{\left|\sqrt{u_e^2 + w_e^2} - \sqrt{u^2 + w^2}\right|}{\sqrt{u_e^2 + w_e^2}} \cong 0.01$$

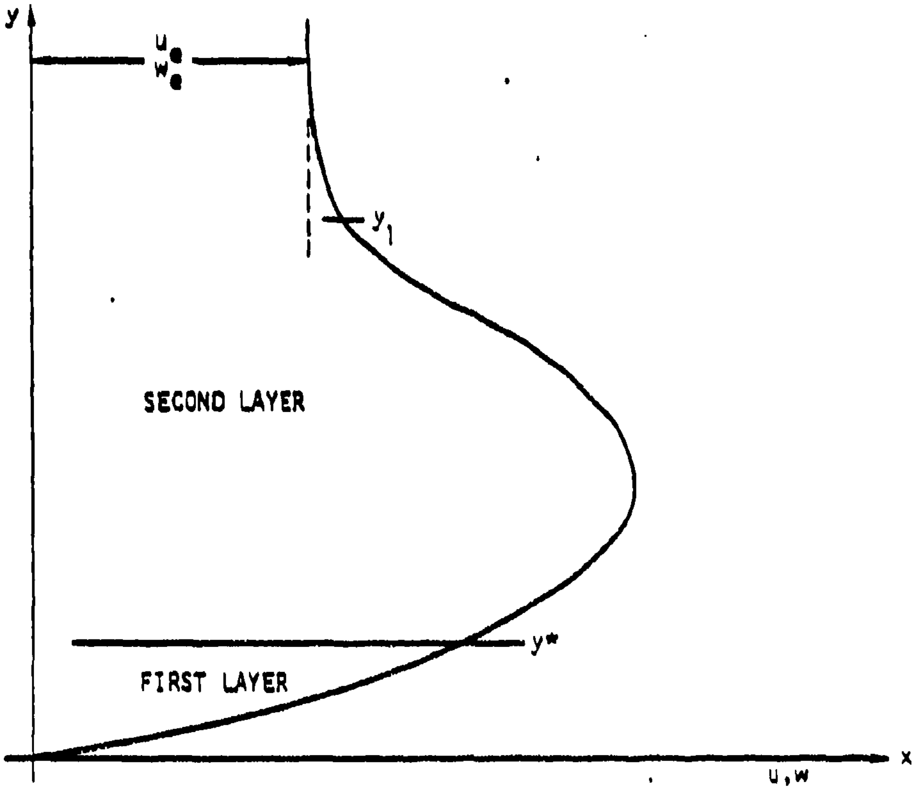


Figure 4. Two-Layer Eddy Viscosity Model

and y^* is obtained by imposing continuity in $\bar{\varepsilon}$ at $y = y^*$. This yields $y^* = \frac{0.125}{0.435} y_1$. Two-layer eddy viscosity models similar to Eq. (10) have been employed by Ramaprian[7] for two-dimensional wall jets which give good comparisons with experiments. In these cases, the term w_y appearing inside the square root in Eq. (10) is zero.

With the concept of eddy viscosity and with the previous transformed variables, it can be shown that the system of Eqs. (1) through (4) can be written as

x-Momentum Equation

$$\begin{aligned}(bf'')' &+ P_1 ff'' + P_2[1 - (f')^2] + P_5[1 - f'g'] \\ &+ P_6 f''g + P_8[1 - (g')^2] \\ &= xP_{10}\left[f' \frac{\partial f'}{\partial x} - f'' \frac{\partial f}{\partial x} + P_7\left(g' \frac{\partial f}{\partial z} - f'' \frac{\partial g}{\partial z}\right)\right] .\end{aligned} \tag{11}$$

z-Momentum Equation

$$\begin{aligned}(bg'')' &+ P_1 fg'' + P_4(1 - f'g') + P_3[1 - (g')^2] \\ &+ P_6 gg'' + P_9[1 - (f')^2] \\ &= xP_{10}\left[f' \frac{\partial g'}{\partial x} - g'' \frac{\partial f}{\partial x} + P_7\left(g' \frac{\partial g'}{\partial z} - g'' \frac{\partial g}{\partial z}\right)\right]\end{aligned} \tag{12}$$

$$\eta = 0 \qquad f = g = g' = g' = 0 \tag{13a}$$

$$\eta = \eta_\infty \qquad f' = g' = 1 . \tag{13b}$$

Here the primes denote differentiation with respect to η, and

$$b = 1+\varepsilon^+ \qquad \varepsilon^+ = \varepsilon/\nu \qquad f' = u/u_e \qquad g' = w/w_e \ . \tag{14a}$$

The coefficients P_1 to P_{10} are functions of u_e, w_e, h_1, h_2, K_1, and K_2 and are given by the following formulas:

$$P_1 = (M+1)/2 - s_1K_2 \qquad P_2 = M \qquad P_3 = R$$

$$P_4 = \left(\frac{u_e}{w_e}\right)Q - s_1K_2 \qquad P_5 = \frac{w_e}{u_e}(N - s_1K_1)$$

$$P_6 = R + \frac{w_e}{2u_e}\left(\frac{1}{h_1}\frac{\partial s_1}{\partial z} - N\right) - \left(\frac{w_e}{u_e}\right) s_1K_1$$

$$P_7 = \frac{h_1}{h_2}\frac{w_e}{u_e} \qquad P_8 = \left(\frac{w_e}{u_e}\right)^2 s_1K_2$$

$$P_9 = \left(\frac{u_e}{w_e}\right) s_1K_1 \qquad P_{10} = \frac{s_1}{xh_1} \tag{14b}$$

$$M = \frac{s_1}{u_eh_1}\frac{\partial u_e}{\partial x} \qquad N = \frac{s_1}{u_eh_2}\frac{\partial u_e}{\partial z}$$

$$Q = \frac{s_1}{u_eh_1}\frac{\partial w_e}{\partial x} \qquad R = \frac{s_1}{u_eh_2}\frac{\partial w_e}{\partial z} \ .$$

In order to solve eqs. (11) through (13), initial conditions are required at a starting plane. In the case of the boundary layer problem, the initial conditions at x=0 and z=0 planes are obtained by solving the limiting form of Eqs. (11) and (12). For a wall jet, initial velocity profiles are prescribed at some downstream $x=x_0$ plane and along the $z=0$ plane, "attachment line" equations are solved. The attachment line equations are obtained by differentiating the z-momentum equation with respect to z and setting

$$w = \frac{\partial p}{\partial z} = \frac{\partial u}{\partial z} = \frac{\partial v}{\partial z} = \frac{\partial^2 w}{\partial z^2} = 0 \; .$$

The resulting attachment line equations valid at the $z=0$ plane are

$$(bf'')' + P_1 ff'' + P_2[1 - (f')^2] + P_3 gf'' = xP_{10}\left[f' \frac{\partial f'}{\partial x} - f'' \frac{\partial f}{\partial x}\right] \qquad (15)$$

$$(bg'')' + P_1 fg'' + P_4(1 - f'g') + P_3[1 - (g')^2]$$

$$+ P_3 gg'' = xP_{10}\left[f' \frac{\partial g'}{\partial x} - g'' \frac{\partial f}{\partial x}\right] . \qquad (16)$$

Here, g' is defined as w_z/w_{e_z} , and its definition corresponds to L'Hospital's rule applied to the expression for g' used previously.

4.0 Finite Difference Equations

First, reduce the system (11)-(12) to the first order system

$$f' = u \qquad (17)$$

$$u' = v \qquad (18)$$

$$g' = w \qquad (19)$$

$$w' = t \qquad (20)$$

$$(bv)' + P_1 fv + P_2(1 - u^2) + P_5[1 - uw]$$

$$+ P_6 vg + P_8[1 - w^2] = xP_{10}\left[u \frac{\partial u}{\partial x} - v \frac{\partial f}{\partial x} + P_7 \left(w \frac{\partial u}{\partial z} - v \frac{\partial g}{\partial z}\right)\right] \quad (21)$$

$$(bt)' + P_1 ft + P_4(1 - uw) + P_3(1 - w^2) + P_6 gt$$

$$+ P_9(1 - u^2) = xP_{10}\left[u \frac{\partial w}{\partial x} - t \frac{\partial f}{\partial x} + P_7 \left(w \frac{\partial w}{\partial z} - t \frac{\partial g}{\partial z}\right)\right] . \quad (22)$$

Using the notation of Ref. 1 associated with the box method described there and in Refs. 3, 4, 5, and 9, we let

$$x_0 = \text{constant} \qquad x_n = x_{n-1} + k_n \qquad n = 1,2,\ldots,N$$

$$z_0 = 0 \qquad z_i = z_{i-1} + r_i \qquad i = 1,2,\ldots,I$$

$$\eta_0 = 0 \qquad \eta_j = \eta_{j-1} + h_j \qquad j = 1,2,\ldots,J .$$

Then, using the box method, we have

$$\frac{f_j^{n,i} - f_{j-1}^{n,i}}{h_j} = u_{j-1/2}^{n,i} \quad (23)$$

$$\frac{u_j^{n,i} - u_{j-1}^{n,i}}{h_j} = v_{j-1/2}^{n,i} \quad (24)$$

$$\frac{g_j^{n,i} - g_{j-1}^{n,i}}{h_j} = w_{j-1/2}^{n,i} \quad (25)$$

$$\frac{w_j^{n,i} - w_{j-1}^{n,i}}{h_j} = t_{j-1/2}^{n,i} . \quad (26)$$

We use the notation

$$\bar{P} = P^{n-1/2,i-1/2}_{j-1/2} \equiv P^{n-1/2}_{i-1/2}$$

and

$$\bar{v}_j = \frac{1}{4}\left(v^{n,i}_j + v^{n,i-1}_j + v^{n-1,i-1}_j + v^{n-1,i}_j\right)$$

$$\bar{u}_n = \frac{1}{2}\left(u^{n,i}_{j-1/2} + u^{n,i-1}_{j-1/2}\right)$$

$$\bar{u}_i = \frac{1}{2}\left(u^{n,i}_{j-1/2} + u^{n-1,i}_{j-1/2}\right) .$$

Equation (21) becomes, with the box centered at $(x_{n-1/2}, z_{i-1/2}, \eta_{j-1/2})$

$$\begin{aligned}(\bar{b}_j\bar{v}_j - \bar{b}_{j-1}\bar{v}_{j-1})/h_j \\ = -\bar{P}_1(\overline{fv})_{j-1/2} - \bar{P}_2(1 - \bar{u}^2_{j-1/2}) - \bar{P}_5(1 - \bar{u}_{j-1/2}w_{j-1/2}) \\ - \bar{P}_6(\overline{vg})_{j-1/2} - \bar{P}_8(1 - \bar{w}^2_{j-1/2}) \\ + x_{n-1/2}\bar{P}_{10}\left\{u_{j-1/2}\frac{(\bar{u}_n - \bar{u}_{n-1})}{k_n} - \bar{v}_{j-1/2}\frac{(\bar{f}_n - \bar{f}_{n-1})}{k_n}\right. \\ \left. + \bar{P}_7\left[\bar{w}_{j-1/2}\frac{(\bar{u}_i - \bar{u}_{i-1})}{r_i} - \bar{v}_{j-1/2}\frac{(\bar{g}_i - \bar{g}_{i-1})}{r_i}\right]\right\} . \qquad (27)\end{aligned}$$

Equation (22) and the attachment line equations (13)-(14) are discretized similarly. Details of the procedure are given in Ref. 9.

The solution procedures involve the following steps:

(1) Solve the attachment line equations (15)-(16) with boundary conditions (13) at $x = x_1$ and $z = 0$ assuming initial conditions on $x = x_0$.

(2) March in the z-direction along the plane $x = x_1$ and solve equations (17)-(22) with boundary conditions (13) for the unknowns (f,u,v,g,w,t).

(3) Repeat steps (1) and (2) for the next x-plane, $x = x_2$, and so on.

The most efficient way to solve the finite difference equations is to use a pseudo-Newton's relaxation scheme. These equations may be written as a system of nonlinear algebraic equations by writing

$$\underset{\sim}{\phi}(\underset{\sim}{u}) = 0$$

where

$$\underset{\sim}{u} = (f_j^{n,i},\ u_j^{n,i},\ v_j^{n,i},\ g_j^{n,i},\ w_j^{n,i},\ t_j^{n,i})_{j=0}^{J} \ .$$

Then, the relaxed Newton's method becomes

$$\frac{\partial\phi}{\partial \underset{\sim}{u}}^{(\nu-1)} \delta \underset{\sim}{u}^{(\nu-1)} = - \underset{\sim}{\phi}\left(\underset{\sim}{u}^{(\nu-1)}\right) \tag{28a}$$

$$\underset{\sim}{u}^{(\nu)} = \underset{\sim}{u}^{(\nu-1)} + \omega\delta\underset{\sim}{u}^{(\nu-1)} \tag{28b}$$

for $\nu = 1,2,\ldots$.

The method is said to have converged when

$$\|\delta \underset{\sim}{u}^{(\nu-1)}\|_\infty \leq \varepsilon \text{ (a prescribed error tolerance)} .$$

We call Eq. (28) a pseudo-Newton's method because we linearize the b terms in Eqs. (21) and (22) by evaluating them at $\nu-2$ before computing the Jacobian matrix, $\partial\underset{\sim}{\Phi}/\partial\underset{\sim}{u}$. Consequently, this algorithm will not be quite quadratically convergent. We, therefore, employ relaxation ($\omega \neq 1$) to accelerate it. Remarkably, underrelaxation ($\omega < 1$) works very well, while overrelaxation ($\omega > 1$) diverges. Values of ω of 0.5, 0.6, 0.7, 0.8, and 0.9 all give good results with $\omega = 0.7$ found to be the overall best for some of our computational experiments.

An important feature of Keller's box method is that the Jacobian matrix can be put into block tridiagonal form and very efficient elimination schemes can be employed for solving Eq. (28a).

Minor Difficulties with the Numerical Algorithm

When starting at $x = 1$ with supplied completely merged wall jet velocity profiles described in Fig. 5, unnatural oscillations developed in the solution. This difficulty was eliminated completely by employing the following "trick." The first 10 mesh points in the x-direction were set at $k_n = 10^{-4}$. For the first five planes in the x-direction and all points in the z-direction in these planes, an average value was used for past points, i.e.,

$$f_j^{n-1,i} = 0.5\left(f_j^{n-1,i} + f_j^{n-2,i}\right) \quad , \quad f_j^{n,i-1} = 0.5\left(f_j^{n,i-1} + f_j^{n,i-2}\right) ,$$

and

$$f_j^{n-1,i-1} = 0.5\left(f_j^{n-1,i-1} + f_j^{n-2,i-1}\right) .$$

SC79-3622

EXTERNAL VELOCITY

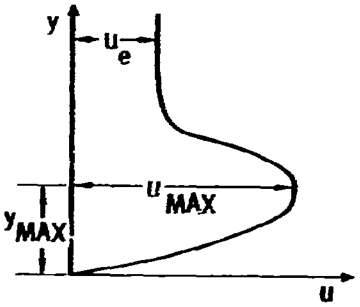
$$u_e = u_o x^{-n} \cos\theta$$

$$w_e = u_o x^{-n} \sin\theta$$

$$\theta = C z (z_{TIP} - z) \frac{\pi}{2}$$

INITIAL PROFILE

$$\frac{u}{u_e} = \begin{cases} C_2 (e^{-y} - 1) + C_3 y & y \leqslant y_{MAX} \\ 1 + \left(\frac{u_{MAX}}{u_e} - 1 \right) e^{-0.1 (y - y_{MAX})^2} & y \geqslant y_{MAX} \end{cases}$$



$$\frac{w}{w_e} = \text{SAME AS } \frac{u}{u_e} \text{ PROFILE}$$

- ● SMALL CROSS FLOW IS ACHIEVED BY SETTING C IN θ EXPRESSION VERY SMALL (10^{-15})
- ● $\frac{u_{MAX}}{u_e}, \frac{w_{MAX}}{w_e}$ CAN BE PRESCRIBED AS A FUNCTION OF z

Figure 5. Initial and Boundary Conditions

Beginning with the sixth x-plane, the averaging was eliminated (the standard algorithm was employed). At the eleventh x-plane a geometric mesh-stretching algorithm of the following form was used:

$$k_n = 1.2k_{n-1} \quad , \quad n = 11,12,13,\ldots .$$

No such stretching has been employed in the z-direction, but in the future it may also be required for rapidly changing profiles. It should be noted that our averaging algorithm was required in both the x and z directions to remove all oscillations.

A mesh refinement algorithm is used which adds or deletes points depending on the relative local variation in the truncation error of the difference equations. Roughly 80 grid points in the η-direction and 11 grid points in the z-direction are employed.

5.0 Results

Computations were performed on the Berkeley CDC 7600 machine. A typical calculation required about 6 minutes of CPU time. Figure 5 indicates the external and initial velocity distributions which have been used as a basis for our calculations. The parameter θ was introduced as shown to vary the initial cross flow while keeping the total velocity constant as a rough simulation of a fixed supply of engine mass flow. The velocity profile was selected to have a characteristic fully developed character associated with turbulent wall jet flows. Also, the surface shape is chosen planar for simplicity in the ensuing analysis. Future aspects of this effort should consider the "eating up" of the potential core which is assumed to occur upstream of the initial station of this analysis. The parameters C_2 and C_3 were chosen to provide slope and value continuity of the profile at $y = y_{max}$. For $y > y_{max}$ the profile has a half Gaussian character associated with a free jet. For $y < y_{max}$ the profile has a boundary layer character. In the examples, the u and w initial profiles were assumed to be identical. Moreover, the θ

distribution was selected to be qualitatively similar to that observed by rake surveys representative of the XFV-12A. The zero cross flow case was achieved by setting C to 10^{-15}.

Figure 6 demonstrates decay of the peak velocity with the standardized distributions of Fig. 5, with and without cross flow. It is evident from the figure that initial cross flow has a dramatic effect on enhancing the decay of the maximum velocity. In the calculations, the decay exponent n in the external velocities is assumed as 1/2, roughly in accord with a value obtained from a two-dimensional line sink simulating inflow originally proposed by G.I. Taylor.[10] Both streamwise increasing and decreasing cross flow cases are shown. In the line sink model, the inflow at each position $x = x_0$ along the jet boundary is determined by streamwise rate of change of mass flux according to conservation of mass applied on a rectangular control surface in the jet layer. From an observation point P in the external flow which is assumed inviscid and irrotational, the velocity potential can therefore be represented as an isolated sink whose intensity is proportional to the inflow at $x = x_0$. The cumulative effect at P of all such sinks at $x = x_1$ is obtained by a superposition integral of all these contributions giving a line sink representation. In a more realistic model, these external velocity distributions should be corrected for three-dimensionality and elliptic interaction with the wall jet. A calculation of this type would be a more accurate representation than the present approach of planform and surface curvature effects. These developments are strongly recommended for future application. In this connection, we recognize that the present means of simulating taper, sweepback, and spanwise pressure gradients is solely through cross flow adjustment.

The three-dimensional inviscid potential ϕ can be characterized by surface sink distribution of the form (see Fig. 7)

$$\phi(x,y,z) = \frac{1}{4\pi}\iint_S \frac{\sigma(\xi,\zeta)d\xi d\zeta}{\sqrt{(x-\xi)^2 + y^2 + (z-\zeta)^2}} \qquad (29)$$

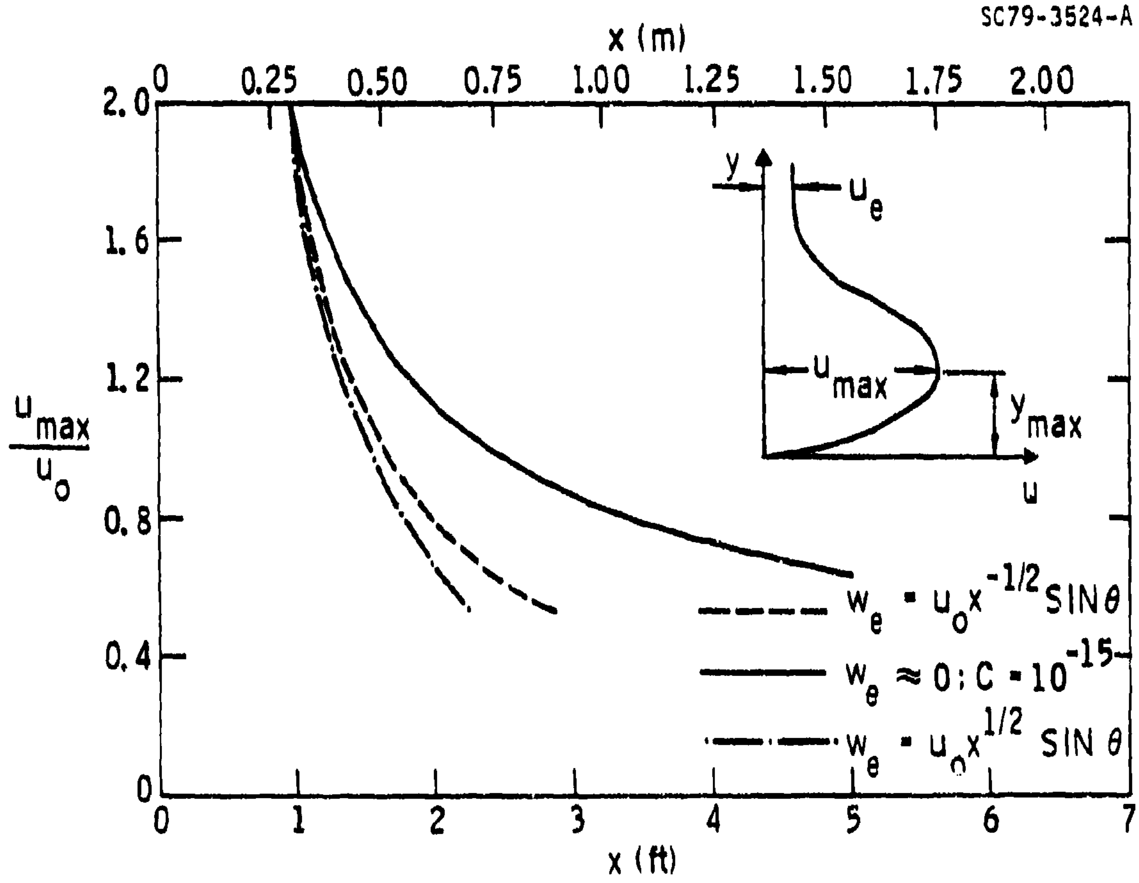


Figure 6. Effect of Cross Flow on Jet Growth, $\frac{u_{max}}{u_e} = 2 - |.5-z|$, $u_e = u_o x^{-1/2} \cos\theta$, $\theta = \frac{\pi}{2} z(z_{tip} - z)$, Standard Initial Profile (See Fig. 5), $z = z_1 = z_{tip}/2$.

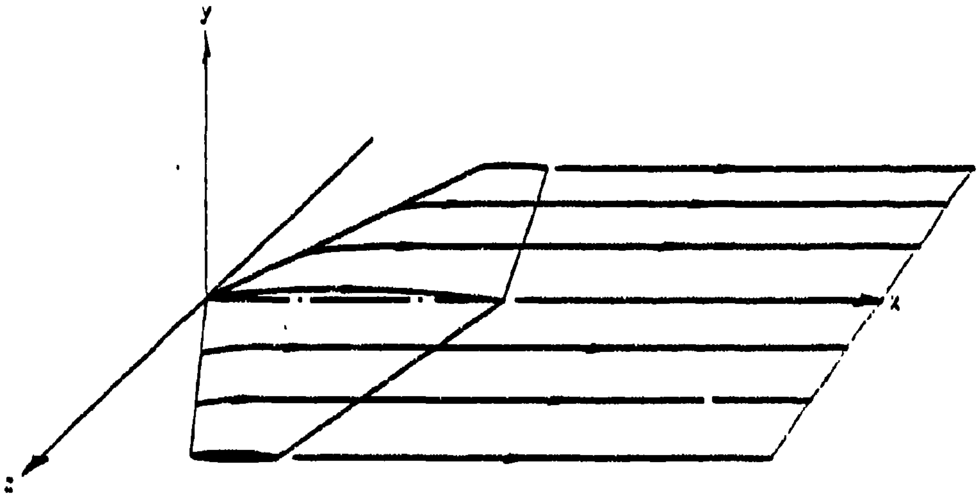


Figure 7. Upper Surface Blown Thrust Augmented Wing (TAW)

where S the area of integration refers to the total jet area on and off the wing. The quantity σ is the sink strength obtained by matching with an "outer limit" of the second order solution for the velocity normal to the body appearing in the viscous inner wall jet solution. Generalizing the previous concepts, Eq. (29) is obtained by superposition of a surface distribution of elementary sink solutions of the form $\phi = \frac{1}{4\pi R}$ where $R^2 = (x-\xi)^2 + y^2 + (z-\zeta)^2$. Note that the surface distributions do not interact in the sense that they produce no ϕ_y at locations other than their own (ξ,ζ). Hence $\phi_y \sim$ local inflow analogous to that previously described for the two-dimensional case. The quantity σ for two-dimensional boundary layers is analogous to the streamwise gradient of the displacement thickness $\delta'(x)$. To include lifting surface effects, a surface doublet or vortex distribution should be added to Eq. (29). The local vortex strength can also be determined by matching.

The inflow velocity related to the sink intensity σ in Eq. (29) is in turn a function of the entrainment. This quantity is also significant from the standpoint of the tradeoff between skin friction, BLC, and rapid acceleration of the secondary in compact three-dimensional thrust augmenting ejectors such as those employed on the XFV-12A and upper surface blowing.

In Fig. 8, the comparison between cross flow and the absence of it gives the indicated nominal entrainment variations with streamwise distance where nominal entrainment Q is defined in the figure. In spite of the appreciable increase in decay of the maximum value of u shown in Fig. 6, and resultant shear stress in Fig. 9, only a slight difference in entrainment quantity and rate is shown in Fig. 8. The difference in maximum velocities which are similar for w, the spanwise component, are presumably related to the enhanced dissipation associated with cross flow and that implied by the eddy viscosity model. The lack of a corresponding increase in entrainment rate may be due to nonlinear compensating effects built into the turbulence model and cannot be readily explained on an intuitive basis at this time. In this connection, other calculations should be performed for which the streamwise component of the

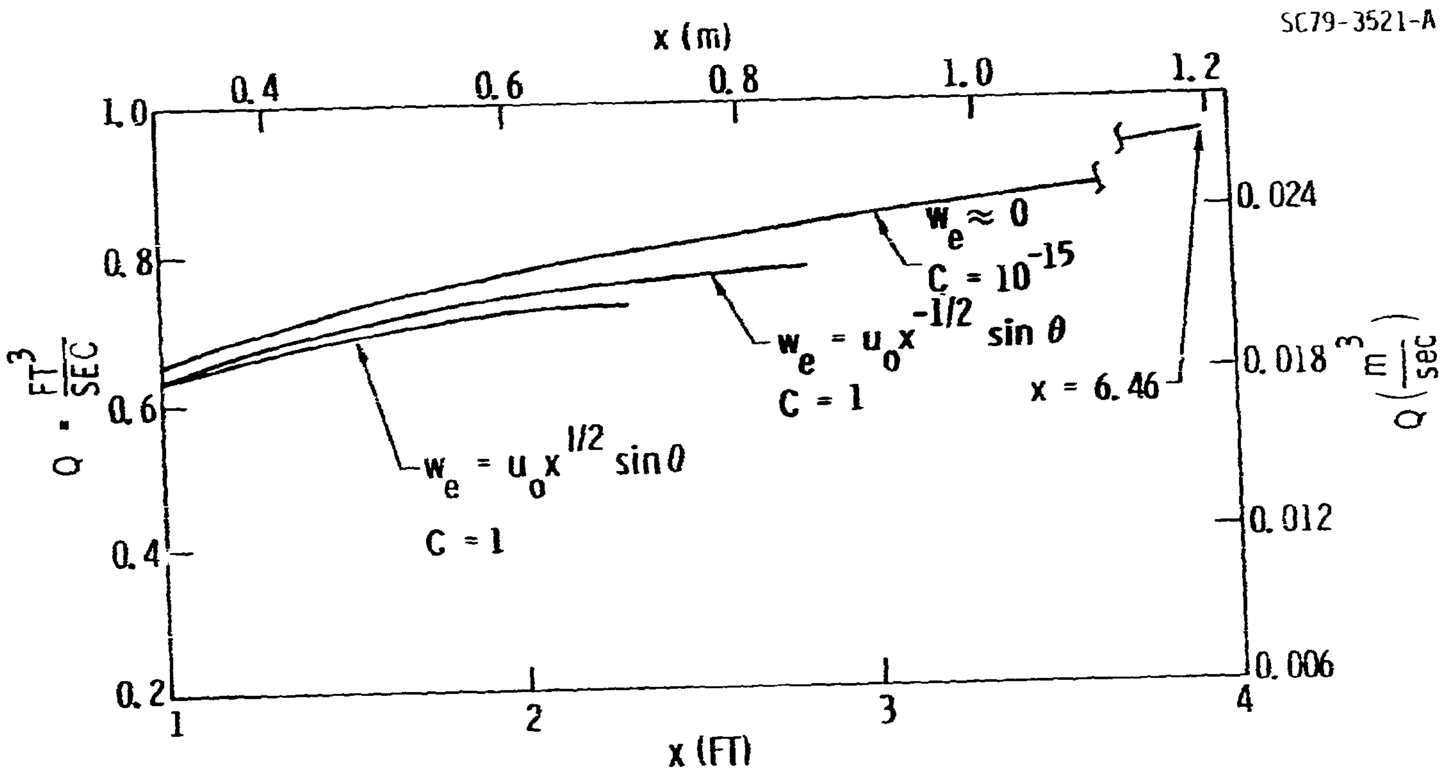


Figure 8. Effect of Cross Flow on Nominal Entrainment, Q with $u_e = u_o x^{-1/2} \cos\theta$,

$$Q = \int_0^\infty \int_0^{z_{tip}} u dy dz, \ \theta = C\frac{\pi}{2} z(z_{tip} - z), \ \frac{u_{max}}{u_e} = 2 - |.5 - z|,$$

(Standard Initial Profile) $z = z_1 = z_{tip}/2$.

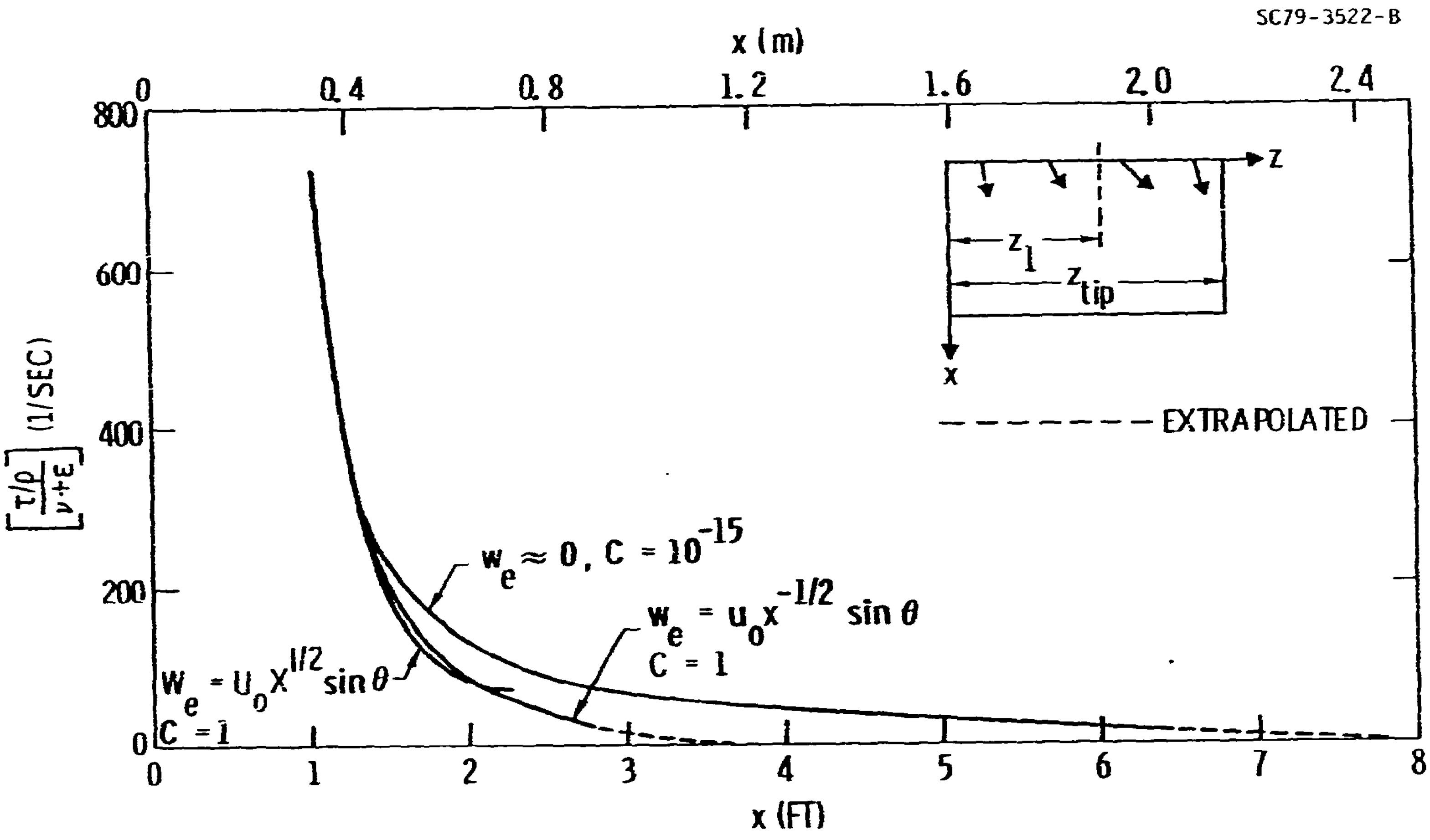


Figure 9. Effect of Cross Flow on Reduced Shear Stress, $u_e = u_o x^{-1/2} \cos\theta$, $\theta = C\frac{\pi}{2}z(z_{tip} - z)$, Standard Initial Profile, $(z = z_1 = z_{tip}/2)$, $\frac{u_{max}}{u_e} = 2 - |.5-z|$.

initial velocity is held fixed rather than its overall magnitude on introduction of cross flow. Furthermore, trends involving an increase in decay rate of peak velocity with an increase in entrainment rate that occur in two-dimensional free jets based on conservation laws and similitude must be reassessed for non-similar three-dimensional wall jets such as those considered herein. Here, the variable streamline direction through the jet layer must be considered as well as the peak of the resultant velocity q_{max} rather than those of the individual velocity components. If s is the local streamline direction on the wall, under certain circumstances, $\frac{\partial q_{max}}{\partial s}$ may become more negative with increasing Q. It should be noted that the expression for entrainment Q given in Fig. 8 assumes that $w = 0$ at the tip $z = z_{tip}$. If this is not the case, an additional term must be added to this relation. As in Figs. 9 and 10, qualitatively similar behavior is obtained for the case in which $w \sim x^{1/2}$.

Associated with the previous results, Fig. 10 shows the effect of cross flow on jet spreading rate related to y_{max}. As previously, only small differences are indicated for the cases considered. In Fig. 11, however, an important upstream movement of the nominal separation line is indicated with the introduction in cross flow. Here, nominal separation is defined to occur where $\tau = 0$. A more pertinent definition is $\frac{\partial q_n}{\partial y} = 0$ where q_n is the velocity component normal to an envelope of the surface streamlines. Implementation of the latter definition is envisioned in follow-on studies involving primary/secondary entrainment interactions. This result is significant with respect to penalties associated with taper and sweep in three-dimensional ejector diffusors.

In Fig. 12, another important consequence of cross flow is examined in connection with the surface streamline pattern. In the figure, two cases are compared involving differing amounts of cross flow. Lines such as AB and A'B' represent typical streamlines for the two different cross flow cases in which the downstream direction is in the direction of the arrows in the rectangular area corresponding roughly to the planform of a

SC79-3523-A

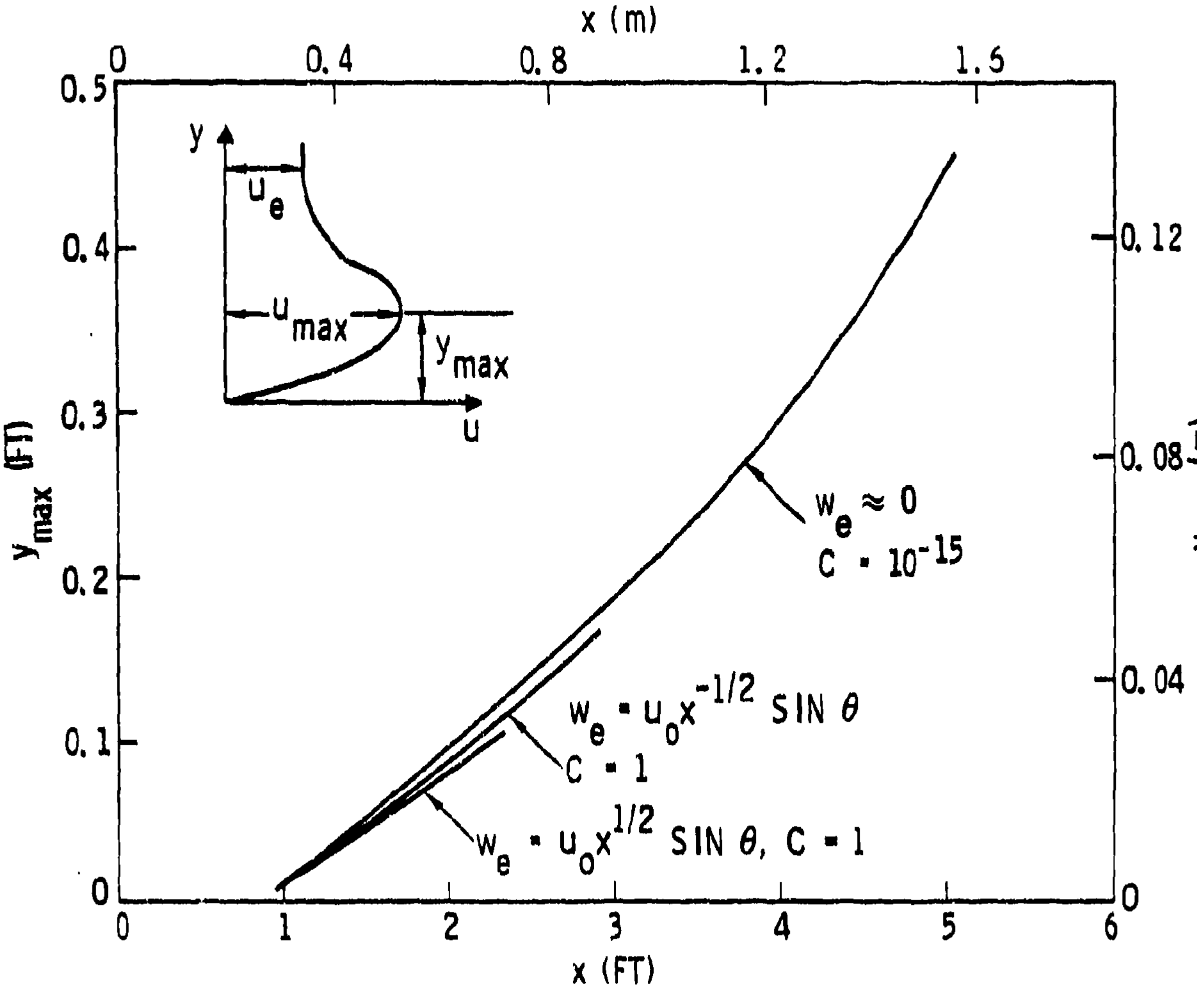


Figure 10. Effect of Cross Flow on Jet Spreading, $u_e = u_o x^{-1/2} \cos \theta$, $\theta = C\frac{\pi}{2}z(z_{tip} - z)$, Standard Initial Profile at Midspan $(z = z_{tip}/2)$, $\frac{u_{max}}{u_e} = 2 - |.5-z|$.

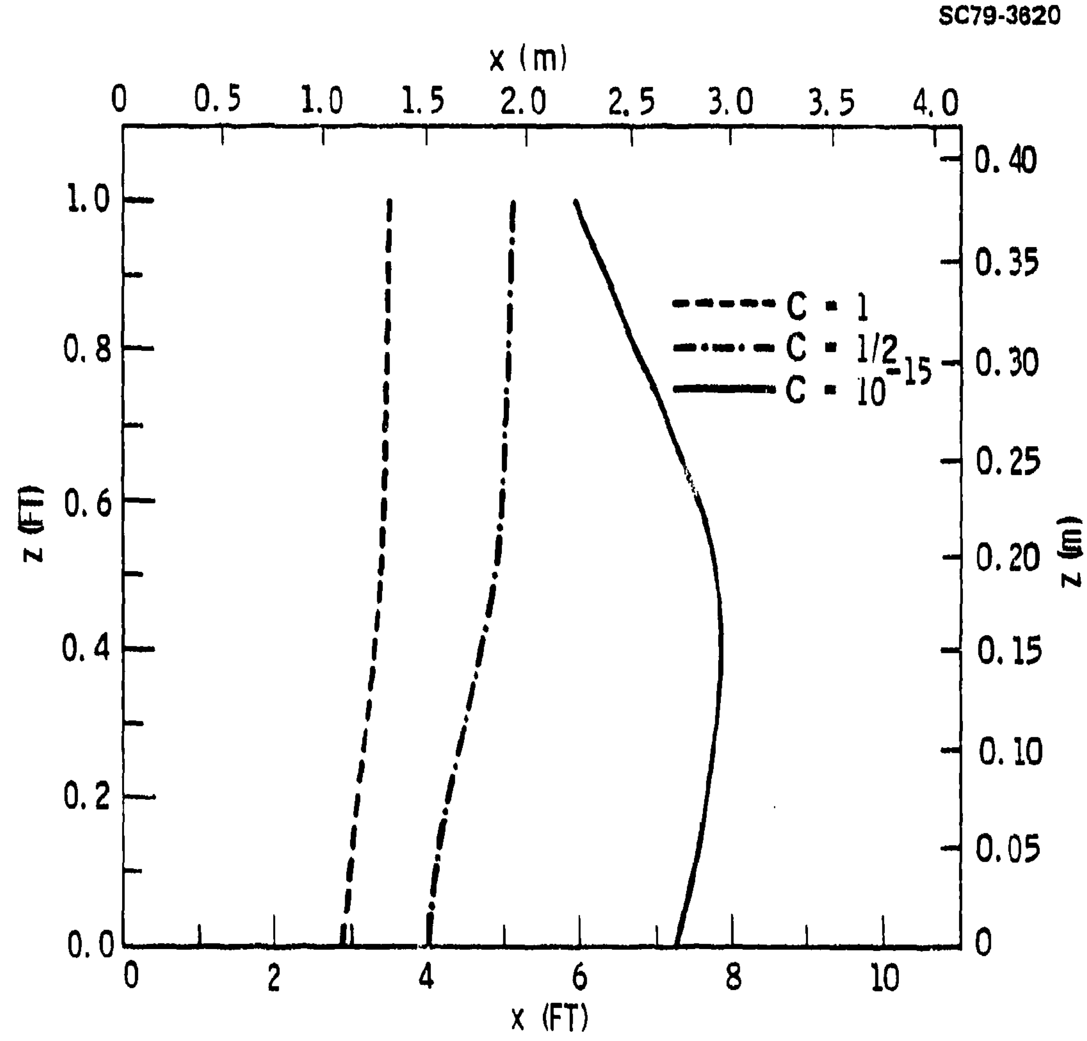


Figure 11. Effect of Cross Flow on the Locus of Nominal Separation, $u_e = u_o x^{-1/2} \cos \theta$, $w_e = u_o x^{-1/2} \sin \theta$, $\theta = C\frac{\pi}{2}z(z_{tip} - z)$, $\frac{u_{max}}{u_e} = 2 - |.5-z|$.

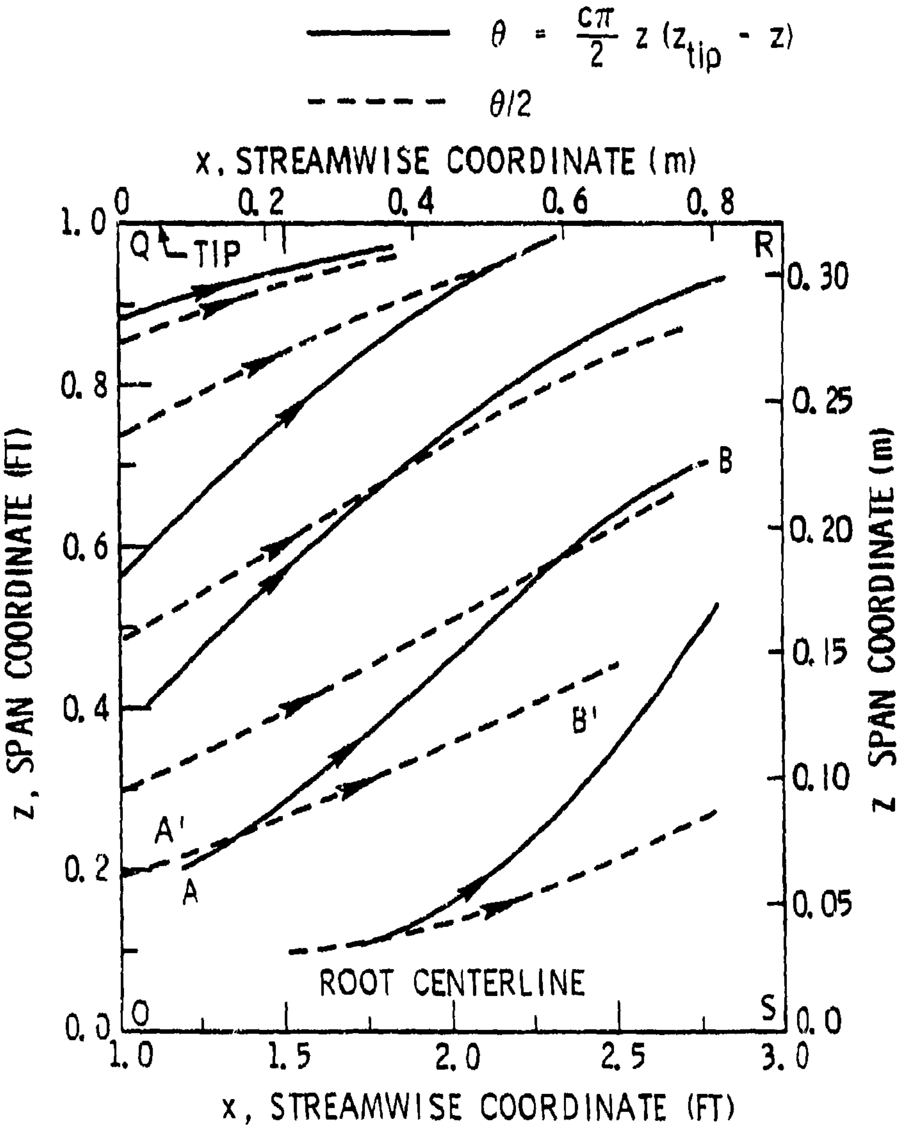


Figure 12. Cross Flow Effect on Jet "Shrink" and "End Wall Pullaway," $u_e = u_o x^{-1/2} \cos\theta$, $w_e = u_o x^{-1/2} \sin\theta$, $\frac{u_{max}}{u_e} = 2 - |.5-z|$.

rectangular wing. In this interpretation, OQ would be the leading edge, RS the trailing edge and QR its tip. Significant enhancement in downstream streamtube contraction is obvious with increase in cross flow. This contraction could presumably lead to end wall separation of the type observed in typical short ejectors. In Fig. 13, a similar picture is indicated for the increasing $w \sim x^{1/2}$ cross flow case. Although some qualitative similarity exists for the decaying case, it is evident that the possibility of a separation surface streamline envelope exists in the pattern of the streamlines.

6.0 Conclusions

A class of cases was investigated roughly possessing initial flow angularity and adverse pressure gradients prototypic of those on the blown surfaces of typical propulsive lift systems such as the Navy/Rockwell XFV-12A thrust augmented wing. Results obtained from the computational model indicate that if the initial total velocity is kept fixed, then the introduction of the cross flow enhances the freestream decay rate of the peak of the velocity component in the freestream direction. In addition, the entrainment quantity and its rate decrease with increased cross flow. The three-dimensional phenomena not only influence the effect of taper on the boundary layer control characteristics of a Coanda flap, but also indicate a "jet shrink" which could be a mechanism promoting end-wall separation. To our knowledge, our model is the first to quantify such trends. Both should be considered in the design of any propulsive lift system. Finally, the effect on the prescribed external adverse pressure gradient in the presence and absence of cross flow has also been examined. From the limited results, the spanwise separation line moves progressively further upstream with increasing cross flow.

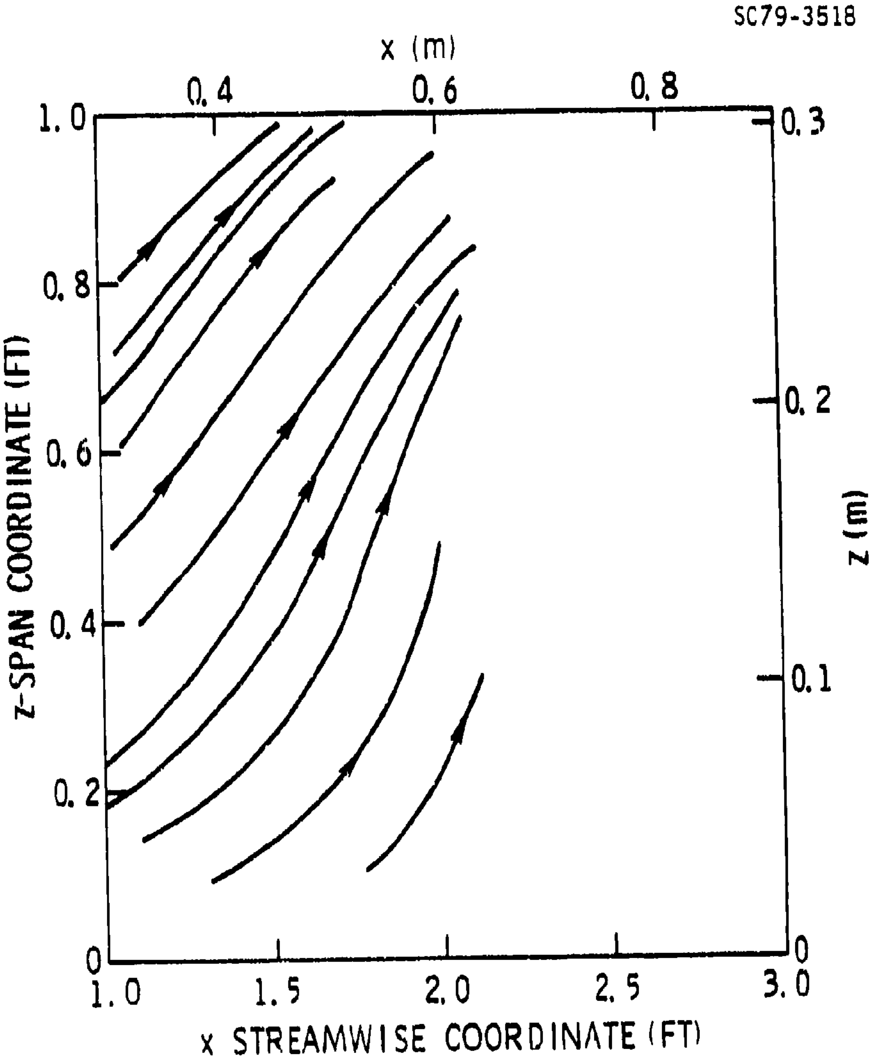


Figure 13. Cross Flow Effect on Jet "Shrink" with Streamwise Increase in Cross Flow, $u_e = u_o x^{-1/2} \cos \theta$, $w_e = u_o x^{1/2} \sin \theta$, $\frac{u_{max}}{u_e} = 2 - |0.5 - z|$, $\theta = \frac{c\pi}{2} z(z_{tip} - z)$

7.0 References

1. Malmuth, N. and Szeto, R., "A Computational Model for Three-Dimensional Incompressible Small Cross Flow Wall Jets," NADC Final Report 76-105-30, December 15, 1977.

2. Reichardt, H., "Gesetzmassigbeiten der Frein Turbulenz," VDI-Forshungsheft, 414 (1942), 2nd ed. (1951).

3. Keller, H.B. and Cebeci, T., "Accurate Numerical Methods for Boundary Layers, I. Two-Dimensional Laminar Flows," Proceedings of the Second International Conference on Numerical Methods in Fluid Dynamics, Lecture Notes in Physics, Springer-Verlag, New York, Vol. 8, 1971.

4. Keller, H.B. and Cebeci, T., "Accurate Numerical Methods for Boundary Layers, II. Two-Dimensional Turbulent Flows," AIAA Journal, Vol. 10, September 1972, pp. 1197-1200.

5. Cebeci, T., "Calculation of Three Dimensional Boundary Layers, II. Three Dimensional Flows in Cartesian Coordinates," AIAA J., Vol. 13, No. 8, p. 1056, 1975.

6. Dvorak, F.A., "Calculation of Turbulent Boundary Layers and Wall Jets Over Curved Surfaces," AIAA J., Vol. 11, No. 4, pp. 517-524, 1973.

7. Ramaprian, B.R., "Turbulent Wall Jets in Conical Diffusers," AIAA J., Vol. 11, No. 12, pp. 1684-1690, September 1973.

8. Wilson, D.J., and Goldstein, R.J., "Turbulent Wall Jets with Cylindrical Streamwise Surface Curvature," Journal of Fluids Engineering, pp. 550-557, September 1976.

9. Murphy, W.D., Shankar, V., and Malmuth, N.D., "Three Dimensional Wall Jets," Quarterly Progress Report No. 2, NADC Contract No. N62269-77-C-0412.

10. Taylor, G.I., J. of Aero. Sci., Vol. 25, 7, pp. 464-465 (1958).

PART II. USERS MANUAL

Although the computer program given in the Appendix was designed to run on the Lawrence Berkeley Laboratory CDC 7600, it may be run on other CDC 6600 or CDC 7600 computers by either removing the plotting routines from the MAIN program or by adapting them to meet the requirements of any other facility.

1. <u>Deck Setup</u> (Berkeley CDC 7600 with plots)

```
Job Card
*NOTAPES
RUN76.
LGO,NL=60000.
DISPOSE,PLOT=PL,R=[SEND PLOTS TO W.D. MURPHY/SCIENCE
+CENTER/ROCKWELL INTERNATIONAL/
+P.O. BOX 1085/
+THOUSAND OAKS, CA. 91360], SC=BKY.
7/8/9

Source program

7/8/9
  CASE 7 UE=1.0/SQRT(X) WE=1.0E-5*Z/SQRT(X) FLAT PLATE
 $INPUTS XSUPLY=.TRUE.,YSUPLY=.TRUE.,ZSUPLY=.TRUE.,OPTPT=.TRUE.$
 ↑
 second column

6/7/8/9
```

Note that the DISPOSE card merely tells Berkeley where to send the plots and may be changed for other users. Also, [= 7/8 punch and] = 0/2/8 punch. For other computer facilities the RUN76 card may be replaced with FTN, and the *NOTAPES and DISPOSE cards should be removed.

2. Estimate of Running Time

The sample case (see the next section and the Appendix) required 389 seconds on the Berkeley CDC 7600. In general, the running time will depend upon the grid selected.

3. Type and Configuration of Computer Used in Program Development

(i) Berkeley CDC 7600 (special feature = Calcomp plotter).

(ii) Any other CDC 6600, CDC 7600, CDC 175, or CDC 176 may be used if the plotting routines are removed from the MAIN program.

4. Name and Level of Programming Language Used in Program

FORTRAN IV.

A. Input-Output Information

Glossary of Input Parameters

XSUPLY Logical variable which is .TRUE. if user supplies the streamwise mesh (x-mesh). Default value = .FALSE..

YSUPLY Logical variable which is .TRUE. if user supplies the η-mesh. Default value = .FALSE..

ZSUPLY Logical variable which is .TRUE. if user supplies the z-mesh. Default value = .FALSE..

XA Real variable. The starting streamwise station. Default value = 0.0. XA may be initialized in subroutine XMESH if XSUPLY=.TRUE..

XB Real variable. The last streamwise station. Default value = 1. XB may be initialized in subroutine XMESH if XSUPLY=.TRUE..

YA Real variable. The left end-point of the η-mesh. Default value = 0.0.

YB Real variable. The right end-point of the η-mesh. Default value = 20. YB may be initialized in subroutine YMESH if YSUPLY=.TRUE..

ZA Real variable. The left end-point of the z-mesh. Default value = 0.0.

ZB Real variable. The right end-point of the z-mesh. Default value = 1. ZB may be initialized in subroutine ZMESH if ZSUPLY=.TRUE..

HKS Real variable initializing the x-mesh in setup of the streamwise mesh and used only if XSUPLY=.FALSE.. Default value = 1.E-5.

FACX Real variable representing the multiplication factor in the setup of the x-mesh if XSUPLY=.FALSE., i.e., X(N)=FACX*X(N-1). for the first few points. Default value = 1.2.

NX Integer variable. The last streamwise station. This may be supplied in subroutine XMESH if XSUPLY=.TRUE.. Default value = 60.

J Integer variable. Total number of η-points which equals the number of internal intervals plus 1. J may be supplied by user in subroutine YMESH if YSUPLY=.TRUE.. Default value = 101.

I Integer variable. Total number of z-points. I may be supplied by user in subroutine ZMESH if ZSUPLY=.TRUE.. Default value = 11. The maximum value is also 11 unless the COMMON BLOCKS in the MAIN program are enlarged.

REFINE Logical variable which is .TRUE. if the η-mesh is to be refined. Default value = .TRUE..

HMAX Real variable. Maximum η-mesh interval permitted in refinement routine. Default value = 2.1.

OPTPT Logical variable which is true if user supplies streamwise stations to be printed on paper. These will also be plotted if the Berkeley 7600 plot routines are being employed. Default value = .FALSE.. User supplies these stations in subroutine PRMESH.

NPRINT Integer variable. Number of uniform streamwise stations at which solution is to be printed ((XB-XA)/NPRINT). Default value = 10.

USUPLY Logical variable which is .TRUE. if user supplies the initial velocity profiles in subroutine PROFLE. Default value = .TRUE.. If these profiles are not supplied, similarity solutions will be generated and used as starting conditions.

R Real variable. Reynolds number. Default value = 1.0/14.216862E-5.

Input

The first card in the input stream contains a TITLE card FORMAT(8A10) describing the problem to be solved. The remaining cards contain NAMELIST data of the variables in the above glossary. The list is defined by

```
NAMELIST /INPUTS/ XSUPLY,YSUPLY,ZSUPLY,XA,....
```

The user must also supply the routines XMESH, YMESH, ZMESH, PRMESH, and PROFLE if the variables XSUPLY, YSUPLY, ZSUPLY, OPTPT, and USUPLY have the value .TRUE., respectively. How these subroutines are written is fully described in the program listing in the Appendix.

In addition, the user must supply the routine PREPP which defines the values of P_1, P_2, ..., P_{10} (equation (14b)) in terms of functions of u_e, w_e, h_1, h_2, K_1, and K_2. In the example given in the Appendix, $u_e = 1/\sqrt{x}$, $w_e = 10^{-5} z/\sqrt{x}$, $h_1 = h_2 = 1$, and $K_1 = K_2 = 0$.

Plots

The program was designed to run on the Berkeley CDC 7600 and to use the Calcomp plotter subroutine package available at Berkeley. It may be run on other CDC computers by either removing the calls to CCNEXT, CCGRID, CCLTR, CCPLOT and CCEND in the MAIN program or by adapting the plot routines to meet the requirements of the new facility.

Graphs are made of the following quantities:

1. y versus u/u_e at z_1, z_4, z_6, z_8, and z_{11}
2. y versus w/w_e at z_1, z_4, z_6, z_8, and z_{11}
3. Shear stress versus x
4. Stream lines
5. u_{max} versus x
6. Jet spreading versus x
7. Entrainment quantity versus x.

The labels for the individual plots are coded in LABEL1, LABEL2, ..., LABEL7 of the MAIN program and may be changed for each individual case study. For these plots the value of u_e must be coded for each new case in the MAIN program and in the subroutine NEWTON where w_e is also required. In the example in the Appendix $u_e = 1/\sqrt{x}$ and $w_e = 10^{-5}\ z/\sqrt{x}$.

Example (sample input deck)

```
    CASE 7 UE=1.0/SQRT(X) WE=1.0E-5*Z/SQRT(X) FLAT PLATE

    $INPUTS XSUPLY=.TRUE.,YSUPLY=.TRUE.,ZSUPLY=.TRUE.,
       OPTPT=.TRUE.$
```

The first card is the title card which is printed on paper, and the second card is a NAMELIST card which implies that the user will supply his own x-mesh, η-mesh, z-mesh, and the streamwise stations to be printed on paper. Examples of these subroutines can be found in the Appendix. All other variables will be assigned their default values.

The initial velocity profiles are coded in subroutine PROFLE and represent the initial conditions given on page 18 of Part I of this report. These initial profiles may easily be changed by following the instructions given in the subroutine.

Output

The title card is printed at the top of the page. If REFINE=.FALSE., the grid points (x-η-z) and y (assuming u_e=1) are printed. This is followed by the initial profiles where F=f, DF=f', DDF=f", G=g, DG=g', and DGG=g". The indices IZ and L represent the subscripts on the z- and η-mesh points, respectively. If REFINE=.TRUE., the program refines the initial mesh and prints the profiles on the new mesh before printing the refined grid. The NAMELIST is printed next.

Every time Newton's method converges for a given streamwise station and z-point the following data is recorded:

1. the number of iterations for convergence (ITER)
2. maximum absolute error (ERROR)
3. element of the solution vector where the maximum error occurs
4. y* defined on page 11 of Part I (YTC)
5. $f''(0)$ (U(3,1))
6. $g''(0)$ (T)
7. b_{10}, b_{20}, b_{30}, b_{40}, b_{50}, b_J where $b = 1+\varepsilon^+$.
8. shear stress (TAU).

After a complete z-plane is swept the following data is printed:

1. entrainment quantity (Q), page 24 of Part I
2. u_{max} at z_1, z_3, z_6, z_8, and z_{10}
3. y_{max} at z_1, z_3, z_6, z_8, and z_{10}.

Note that u_{max} and y_{max} are defined on page 27 of Part I.

The complete solution is printed at those streamwise stations previously specified by the user in PRMESH or at default values. The format for this printing is the same as that for the initial profiles. Plot data will also be loaded into storage vectors at these stations for possible later use by the plotting subroutines in the MAIN program.

Example (sample output listing)

Because of the three-dimensional nature of this problem, a complete listing of the output would be too exhaustive for this report. Therefore, we give below the first page of output that resulted from the sample input deck given earlier.

```
        CASE 6   UE=COS(TH)/SQRT(X)   WE=SIN(TH)/SQRT(X)    FLAT PLATE

NEWTON - ITER = 4 ERROR = 7.08503E-03 AT( 6,  3 )      YTC =  1.96462E+00 U(3,1)  3.71916E+00 1  3.75751E+00
     B=  1.52105E+00  6.03638E+00  8.03161E+00  1.20877E+00  1.09424E+00  1.03168E+00  TAU=  3.11881E+02

       INITIAL PROFILES

IZ   L        F             DF           DDF            G             DG           DDG

 1   1  0.            0.            3.58832E+00  0.            0.            3.58832E+00
 1   2  1.78614E-02   3.43506E-01   3.02764E+00  1.78614E-02   3.43506E-01   3.02764E+00
 1   3  2.77093E-02   4.19928E-01   2.09734E+00  2.77093E-02   4.19928E-01   2.39734E+00
 1   4  4.20615E-02   5.07261E-01   2.74535E+00  4.20615E-02   5.07261E-01   2.74535E+00
 1   5  6.27398E-02   6.05957E-01   2.56908E+00  6.27398E-02   6.05957E-01   2.56908E+00
 1   6  9.22050E-02   7.15928E-01   2.36601E+00  9.22050E-02   7.15928E-01   2.36601E+00
 1   7  1.33723E-01   8.36240E-01   2.13398E+00  1.33723E-01   8.36240E-01   2.13398E+00
 1   8  1.91530E-01   9.64715E-01   1.87144E+00  1.91530E-01   9.64715E-01   1.87144E+00
 1   9  2.70959E-01   1.09743E+00   1.57785E+00  2.70959E-01   1.09743E+00   1.57785E+00
 1  10  3.23279E-01   1.16650E+00   1.41226E+00  3.23279E-01   1.16650E+00   1.41226E+00
 1  11  3.78620E-01   1.22810E+00   1.25414E+00  3.78620E-01   1.22810E+00   1.25414E+00
 1  12  4.40525E-01   1.29261E+00   1.07380E+00  4.40525E-01   1.29261E+00   1.07380E+00
 1  13  5.21740E-01   1.34739E+00   9.03184E-01  5.21740E-01   1.34739E+00   9.03184E-01
 1  14  6.13205E-01   1.40103E+00   7.10556E-01  6.13205E-01   1.40103E+00   7.10556E-01
 1  15  7.07827E-01   1.44225E+00   5.30332E-01  7.07827E-01   1.44225E+00   5.30332E-01
 1  16  7.85277E-01   1.46685E+00   3.94545E-01  7.85277E-01   1.46685E+00   3.94545E-01
 1  17  8.63048E-01   1.48440E+00   2.65799E-01  8.63048E-01   1.48440E+00   2.65799E-01
 1  18  9.43177E-01   1.49527E+00   1.43729E-01  9.43177E-01   1.49527E+00   1.43729E-01
 1  19  1.01494E+00   1.49964E+00   3.92870E-02  1.01494E+00   1.49964E+00   3.92870E-02
 1  20  1.08681E+00   1.49996E+00  -2.92815E-03  1.08681E+00   1.49996E+00  -2.92815E-03
 1  21  1.15869E+00   1.49970E+00  -7.71600E-03  1.15869E+00   1.49970E+00  -7.71600E-03
 1  22  1.23054E+00   1.49922E+00  -1.24932E-02  1.23054E+00   1.49922E+00  -1.24932E-02
 1  23  1.34545E+00   1.49797E+00  -2.00983E-02  1.34545E+00   1.49797E+00  -2.00983E-02
 1  24  1.46023E+00   1.49614E+00  -2.76327E-02  1.46023E+00   1.49614E+00  -2.76327E-02
 1  25  1.57466E+00   1.49373E+00  -3.50702E-02  1.57486E+00   1.49373E+00  -3.50702E-02
 1  26  1.78862E+00   1.48799E+00  -4.81319E-02  1.78862E+00   1.48799E+00  -4.81319E-02
 1  27  1.98547E+00   1.48047E+00  -6.06533E-02  1.98547E+00   1.48047E+00  -6.06533E-02
 1  28  2.22973E+00   1.46924E+00  -7.47787E-02  2.22973E+00   1.46924E+00  -7.47787E-02
 1  29  2.47195E+00   1.45577E+00  -8.77280E-02  2.47195E+00   1.45577E+00  -8.77280E-02
 1  30  3.04261E+00   1.41558E+00  -1.13030E-01  3.04261E+00   1.41558E+00  -1.13030E-01
 1  31  3.37683E+00   1.38727E+00  -1.23802E-01  3.37683E+00   1.38727E+00  -1.23802E-01
 1  32  3.70405E+00   1.35681E+00  -1.31083E-01  3.70405E+00   1.35681E+00  -1.31083E-01
 1  33  4.36565E+00   1.28960E+00  -1.35353E-01  4.36565E+00   1.28960E+00  -1.35353E-01
 1  34  4.99395E+00   1.22359E+00  -1.26860E-01  4.99395E+00   1.22359E+00  -1.26860E-01
 1  35  6.16310E+00   1.11471E+00  -8.80266E-02  6.16310E+00   1.11471E+00  -8.80266E-02
 1  36  6.71084E+00   1.07623E+00  -6.61198E-02  6.71084E+00   1.07623E+00  -6.61198E-02
```

B. Subroutine Description

All subroutines contain many comment cards describing the routine and its function. Our purpose here is not to duplicate all this information but instead to discuss some of the technical details of each routine. An * denotes the routine is user supplied.

SUBROUTINE NAME: BC

PURPOSE: This routine computes the boundary conditions and the Jacobian of the boundary data. See equation (13) of Part I.

DESCRIPTION: The solution vector is written in the form $u = (f,f',f'',g,g',g'')^T$, and the variables A and B denote left and right boundary, respectively. The boundary conditions are assumed to be written in homogeneous form and the left boundary conditions are coded before the right ones. For example, the right boundary condition

$$f'(\eta_\infty) = 1$$

translates into code as

```
G(5) = UB(2)-1.0.
```

SUBROUTINE NAME: BETASV

PURPOSE: This subroutine solves for β in the LU-decomposition of the block tridiagonal Jacobian matrix (equation (28a) in Part I).

DESCRIPTION: The block tridiagonal matrix is written as

$$/A \equiv [B_i, A_i, C_i] \qquad (i=1,2,\ldots,J)$$

where A_i, B_i, and C_i are matrices of order 6.

/A is decomposed as

$$/A \equiv [\beta_i \; I \; 0][0 \; \alpha_i \; C_i]$$

where

$$\alpha_1 = A_1 \qquad (1a)$$

$$\beta_i \, \alpha_{i-1} = B_i \qquad i=2,3,\ldots,J \qquad (1b)$$

$$\alpha_i = A_i - \beta_i \, C_{i-1} \qquad i=2,3,\ldots,J\,. \qquad (1c)$$

For simplicity, we write the LU decomposition of α_i as

$$\alpha_i = \ell_i \, u_i\,. \qquad (2)$$

If this decomposition doesn't exist, then permutation matrices for row and column pivoting can be introduced in equation (2). Assuming p rows in B_i, the solution of β_i in equation (1b) is given by

$$\left.\begin{array}{l} u_{i-1}^T \, \underset{\sim}{y}_k = \underset{\sim}{B}_i^T \; k \\ \ell_{i-1}^T \, \underset{\sim}{\beta}_k^T = \underset{\sim}{y}_k \end{array}\right\} \quad k=1,2,\ldots,p\,. \qquad (3)$$

SUBROUTINE NAME: BLOCK1

PURPOSE: This subroutine decomposes the block tridiagonal Jacobian matrix into LU-form.

DESCRIPTION: The steps in equations (1-3) above are carried out by performing the indicated matrix multiplications and calling routines LUSOLV and BETASV.

SUBROUTINE NAME: BLOCK2

PURPOSE: This routine solves the block tridiagonal system /Ax = b assuming the factorized form.

DESCRIPTION: Write $z \equiv (\underset{\sim}{z}_1, \underset{\sim}{z}_2, \ldots, \underset{\sim}{z}_J)^T$ and $x \equiv (\underset{\sim}{x}_1, \underset{\sim}{x}_2, \ldots, \underset{\sim}{x}_J)^T$ where $\underset{\sim}{z}_\ell = (z_{1\ell}, z_{2\ell}, \ldots, z_{6\ell})^T$ and $\underset{\sim}{x}_\ell = (x_{1\ell}, x_{2\ell}, \ldots, x_{6\ell})^T$. Then BLOCK2 first solves

$$\underset{\sim}{z}_1 = \underset{\sim}{b}_1 \qquad (4a)$$

$$\underset{\sim}{z}_k = \underset{\sim}{b}_k - \beta_k \underset{\sim}{z}_{k-1} \qquad k=2,3,\ldots,J \qquad (4b)$$

and then

$$\alpha_J \underset{\sim}{x}_J = \underset{\sim}{z}_J \qquad (5a)$$

$$\alpha_{\ell-1} \underset{\sim}{x}_{\ell-1} = \underset{\sim}{z}_{\ell-1} - C_{\ell-1} \underset{\sim}{x}_\ell \qquad \ell=J,J-1,\ldots,2\ . \qquad (5b)$$

Equation (5a) is solved by calling USOLVE.

SUBROUTINE NAME: BOX

PURPOSE: This subroutine sets up the block tridiagonal system of equations for the attachment line equation (15-16) of Part I when KASE=2 and the 3-D wall jet equations (23-27) of Part I when KASE=3.

DESCRIPTION: The elements of $[B_i, A_i, C_i]$ are loaded by calling the routines BC, RHSF, and RHSF2D which contain Jacobian information for the boundary conditions and the various difference equations. Previous station information and updating of the turbulence model is included by incorporating the G matrix (see Appendix for the component description). Finally, the equations are rearranged to ensure the first diagonal block is nonsingular.

SUBROUTINE NAME: LUSOLV

PURPOSE: This subroutine decomposes a scalar matrix into LU-form using a mixed pivoting strategy.

DESCRIPTION: See Analysis of Numerical Methods by E. Isaacson and H.B. Keller for a standard discussion of Gaussian elimination (LU decomposition) with pivoting.

SUBROUTINE NAME: MAIN

PURPOSE: This is the main driving program.

DESCRIPTION: The MAIN program initializes solution vectors, sets default values, reads and writes NAMELIST data, and calls subroutines that initialize the grid and the velocity profiles. The marching scheme is called from MAIN as well as all plotting packages.

SUBROUTINE NAME: NETRUN

PURPOSE: This subroutine is used to add or delete η-mesh points in order to improve the smoothness of the solution especially in regions where large boundary layers exist.

DESCRIPTION: The local truncation error of the numerical scheme is approximated at the mid-point of each sub-interval of the initial grid. Points are added or deleted so that this error will remain constant on the whole interval. NETRUN is only called for the first x-point and z-point, if the user requests this option (REFINE=.TRUE.). This new improved grid is used for the entire 3-D region.

SUBROUTINE NAME: NEWTON

PURPOSE: This subroutine solves the 2-D attachment line equation when KASE=2 and the 3-D wall jet equation when KASE=3 by employing the underrelaxed Newton's method (equation (28) of Part I).

DESCRIPTION: The following notation is used:

N=N1 = number of unknowns in the solution vector = 6

NP=N2 = number of boundary conditions defined at the left boundary = 4

NQ=N3 = number of boundary conditions defined at the right boundary = 2.

Previous station information is loaded from subroutine PREPG and the eddy-viscosity from PREPB. After the proper block tridiagonal Jacobian matrix is loaded (KASE=2 or 3), it is solved using BLOCK1 and BLOCK2. Underrelaxation (ω=.7) is performed on this solution, and a maximum absolute or relative (REL=.TRUE.) error between iterations is computed. If this error is greater than EPSERR=10^{-2} (loaded from MAIN), the above steps are repeated; otherwise, plot and print vectors are loaded. u_e and w_e are coded into this subroutine and must be changed for each problem.

SUBROUTINE NAME: OUTPT

PURPOSE: This subroutine writes the solution on paper for the plane X=XN.

DESCRIPTION: The logic is set up so that the initial profile will only be printed once. The x-station is loaded into the plot vector for later use. The array ULAST contains f' and g' for future plots. The solution vector is printed on paper using the format described in the Output section.

SUBROUTINE NAME: PREP

PURPOSE: This routine sets up the initial grid and loads initial conditions into the solution vector. The grid is printed from this subroutine.

DESCRIPTION: Mesh points use default values if XSUPLY, YSUPLY, and ZSUPLY are .FALSE.. For each .TRUE. value a user supplied subroutine is called. Values of x, η, κ, and y are printed where y is computed from u_e at some x station, usually x_F. The user should change u_e with every new case study. If USUPLY=.TRUE., initial conditions are loaded from the user supplied routine PROFLE; otherwise, a similarity solution is generated from initial exponential decaying functions given by statement functions defined by F1(T), F2(T), and F3(T). Grid refinement on either of these initial conditions will be performed if REFINE=.TRUE..

SUBROUTINE NAME: PREPB

PURPOSE: This subroutine models turbulence using the two layers discussed on pages 8 through 11 of Part I.

DESCRIPTION: The values of c_1 and c_2 (TC1 and TC2, respectively) have been set to zero using a DATA STATEMENT but they may be assigned values between zero and three. The program is divided into two parts depending upon whether the attachment line equations (IZ=1) or the 3-D wall jet equations (IZ≠1) are being solved. In each part the value of y* (YTC) is determined, and the quantities $b = 1+\epsilon^+$ in equations (11) and (12) of Part I are computed and stored in the second and sixth rows of the G matrix, respectively. Previous station data is also updated and stored in the third and fourth rows.

SUBROUTINE NAME: PREPG

PURPOSE: This subroutine computes previous station data for the full 3-D wall jet equations.

DESCRIPTION: The actual variables coded in this routine are T (G(3,L)) and S (G(4,L)) which were first derived in Ref. 9 of Part I. The following notation is used for the FORTRAN variables:

$$\text{UHX} = u^{n,i-1}_{j-1/2}$$

$$\text{UHXX} = u^{n-1,i}_{j-1/2}$$

$$\text{UHXXX} = u^{n-1,i-1}_{j-1/2}$$

$$\text{UT} = \tilde{u}_{j-1/2} \quad .$$

SUBROUTINE NAME: PREPG2

PURPOSE: This subroutine computes previous station data for the attachment line equation (equations (15-16) of Part I).

DESCRIPTION: The quantities $T^{n-1}_{j-1/2}$ (G(3,L)) and $S^{n-1}_{j-1/2}$ (G(4,L)) of Ref. 9 are coded in this subroutine. UH denotes the variable $u^{n-1}_{j-1/2}$.

SUBROUTINE NAME: PREPP*

PURPOSE: The variables P_1, P_2, ..., P_{10} given by equation (14) of Part I are coded in this subroutine.

DESCRIPTION: For each new problem the following quantities must be coded into this subroutine:

UE = u_e

UEX = $\partial u_e/\partial x$

UEZ = $\partial u_e/\partial z$

WE = w_e

WEX = $\partial w_e/\partial x$

WEZ = $\partial w_e/\partial z$

WEZX = $\partial^2 w_e/\partial z\partial x$.

The values of P_1, P_2, ..., P_{10} in this routine are written for the special case of a flat plate; however, they may easily be changed to exactly those given by equation (14). Note that when IZ=1, the routine computes the corresponding values of P for the attachment line equations.

SUBROUTINE NAME: PRMESH*

PURPOSE: This routine allows the user to supply the streamwise stations at which solutions are to be printed on paper. These stations will also serve as those where plot vectors will be loaded.

DESCRIPTION: See program listing in the Appendix.

SUBROUTINE NAME: PROFLE*

PURPOSE: This routine sets up the initial velocity profiles.

DESCRIPTION: In the sample case given in the Appendix we have coded the initial conditions described on page 18 of Part I. The quantity u_{max}/u_e varies from 1.5 to 2.0. The first six rows of the USTORE matrix contain the values of (f,f',f'',g,g',g'') as a function of η_ℓ and z_m. If REFINE=.TRUE., the mesh refinement routine alters the η-mesh in order to make the initial conditions smoother between grid points. New initial conditions are computed and printed at the refined points.

SUBROUTINE NAME: RHSF

PURPOSE: This routine computes the quantities on the right-hand side of the equations given by (31-36) in Ref. 9. The Jacobian matrix is also evaluated and stored in the A matrix.

DESCRIPTION: The following notation is used:

UT = $\tilde{u}/4.0$

UBAR = $\bar{u}$

U = current value of $u^{n,i}_{j-1/2}$

UX = pass value $u^{n,i-1}_{j-1/2}$

UXX = pass value $u^{n-1,i}_{j-1/2}$

UXXX = pass value $u^{n-1,i-1}_{j-1/2}$

The values of the right-hand side of equations (31-36) are stored in F(1), F(2), ..., F(6), respectively

SUBROUTINE NAME: RHSF2D

PURPOSE: RHSF2D is similar to RHSF except the attachment line equations given by (38) in Ref. 9 and the corresponding Jacobian matrix are coded.

DESCRIPTION: See RHSF above.

SUBROUTINE NAME: TRUN

PURPOSE: This subroutine computes the local truncation error of the centered-Euler scheme.

DESCRIPTION: See H.B. Keller, "Accurate Difference Methods for Nonlinear Two-point Boundary Value Problems," SIAM J. Numer. Anal., 11 (1974), pp. 305-320.

SUBROUTINE NAME: USOLVE

PURPOSE: This routine solves the scalar matrix equation $Ax = f$ after A has been put in LU-form.

DESCRIPTION: See E. Isaacson and H.B. Keller, *Analysis of Numerical Methods*, J. Wiley & Sons, New York, 1966.

SUBROUTINE NAME: VANDET

PURPOSE: This function computes the determinant of an $n \times n$ Vandermonde matrix for $1 < n < 7$. This routine is called by TRUN in order to compute the local truncation error of the centered-Euler or box scheme.

DESCRIPTION: The Vandermonde determinant is given by

$$V = \begin{vmatrix} 1 & x_0 & \cdots & x_0^n \\ 1 & x_1 & \cdots & x_1^n \\ \cdot & & & \\ \cdot & & & \\ \cdot & & & \\ 1 & x_n & \cdots & x_n^n \end{vmatrix} = \prod_{i>j} (x_i - x_j) .$$

SUBROUTINE NAME: VELMAX

PURPOSE: This subroutine computes u_{max} and y_{max} described on page 18 of Part I.

DESCRIPTION: $u(\eta_j)$ is computed for each mesh point η_j and the maximum value is determined, say $u_{max} = u(\eta^*)$. Then y_{max} is given by $y_{max} = \eta^* \sqrt{\nu x/u_e}$.

SUBROUTINE NAME: WALJET

PURPOSE: This is the main marching routine which calls NEWTON to solve either the 3-D wall jet problem (equations (11-14) of Part I).or the 2-D attachment line equations (equations (15-16) of Part I).

DESCRIPTION: The marching direction is discussed on page 16 of Part I. Unnatural oscillations are damped out by the averaging procedure discussed on pages 17 and 19 of Part I. The OUTPT routine is called at the appropriate point from this subroutine depending upon whether OPTPT is .TRUE. or .FALSE..

If separation ($f''(0) < 0$) occurs, the marching is terminated and the solution at the previous streamwise station is printed.

SUBROUTINE NAME: XMESH*

PURPOSE: This routine allows the user to input his own x-mesh.

DESCRIPTION: See the Appendix for an example and a complete explanation.

SUBROUTINE NAME: YMESH*

PURPOSE: This routine allows the user to input his own η-mesh.

DESCRIPTION: See the Appendix for an example and a complete explanation.

SUBROUTINE NAME: ZMESH*

PURPOSE: This routine allows the user to input his own z-mesh.

DESCRIPTION: See the Appendix for an example and a complete explanation.

APPENDIX

PROGRAM LISTING

```
      PROGRAM MAIN(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,PLOT,          00000010
     X              TAPE98=101,TAPE99=PLOT)                              00000012
C                                                                        00000020
C THIS IS THE MAIN PROGRAM FOR SOLVING THE THREE-DIMENSION INCOM-       00000030
C PRESSIBLE WALL JET EQUATION WITH LARGE CROSS FLOW.  H. B. KELLER*S BOX00000040
C SCHEME (REFERENCE 1) IS USED TO DISCRETIZE THE NONLINEAR SYSTEM OF    00000050
C PARTIAL DIFFERENTIAL EQUATIONS.  THE DISCRETIZED SYSTEM IS THEN SOLVED00000060
C BY NEWTON*S METHOD.                                                    00000070
C THE FIRST CARD IN THE INPUT STREAM IS THE TITLE CARD (FORMAT(8A10)).  00000080
C THE REMAINING DATA IS ENTERED BY NAMELIST ** INPUTS **.  THOSE         00000090
C VARIABLES THAT ARE INPUT PARAMETERS IN THIS NAMELIST ARE ENCLOSED      00000100
C BY ** .... ** AT THE END OF THE DESCRIPTION BELOW.                     00000110
C THE PLOTTING ROUTINES GIVE GRAPHS OF U/UE VS Y, W/WE VS Y, SHEAR       00000120
C STRESS VS X, STREAM LINES, UMAX VS X, JET SPREADING VS X, AND THE      00000130
C ENTRAINMENT QUANTITY VS X.                                             00000140
C                                                                        00000150
C  DESCRIPTION OF MAIN VARIABLES                                         00000160
C         A   MAIN DIAGONAL BLOCKS OF THE BLOCK-TRIDIAGONAL MATRIX       00000170
C             (B A C) OBTAINED FROM LINEARIZATION OF THE FINITE-DIFFERENCE00000180
C             APPROXIMATION OF THE GOVERNING EQUATIONS                   00000190
C       AJA   JACOBIAN MATRIX USED IN SETUP OF THE BLOCK TRIDIAGONAL     00000200
C             MATRIX                                                     00000210
C         B   LOWER DIAGONAL BLOCKS. SEE DESCRIPTION OF VARIABLE A       00000220
C         C   UPPER DIAGONAL BLOCKS. SEE DESCRIPTION OF VARIABLE A       00000230
C  CCFACT     REAL VARIABLES IN THIS COMMON BLOCK ARE USED FOR PLOTTING  00000240
C             ON THE BERKELEY CDC 7600                                   00000250
C  CCPOOL     MORE PLOTTING VARIABLES IN THIS COMMON BLOCK               00000260
C        DU   REAL VECTOR VARIABLE. THE DIFFERENCE BETWEEN TWO NEWTON    00000270
C             ITERATES                                                   00000280
C  EPSERR     REAL CONSTANT VARIABLE, CONVERGENCE CRITERION OF NEWTON#S  00000290
C             METHOD                                                     00000300
C       ETA   REAL VECTOR, ETA MESH POINTS                               00000310
C         F   REAL VECTOR VARIABLE, RIGHT HAND SIDE OF THE BLOCK TRI-    00000320
C             DIAGONAL SYSTEM OF EQUATIONS                               00000330
C     FACX    REAL CONSTANT VARIABLE, MULTIPLICATION FACTOR IN SETUP OF  00000340
C             STREAMWISE MESH, **INPUT, DEFAULT VALUE=1.2**              00000350
C        FF   REAL VECTOR VARIABLE, USED IN SETUP OF BLOCK TRIDIAGONAL   00000360
C             SYSTEM OF EQUATIONS                                        00000370
C         G   REAL VECTOR VARIABLE, USED IN SETUP OF BLOCK TRIDIAGONAL   00000380
C             SYSTEM OF EQUATIONS (THE THIRD AND FOURTH COMPONENTS CONTAIN00000390
C             CONTRIBUTIONS FROM A PREVIOUS X OR Z STATION, AND THE      00000400
C             REMAINING COMPONENTS ARE USED IN THE TWO LAYER TURBULENCE  00000410
C             MIXING LENGTH MODEL).                                      00000420
C       HKS   REAL CONSTANT VARIABLE, INITIAL MESH-SIZE IN SETUP OF      00000430
C             STREAMWISE MESH,**INPUT, DEFAULT VALUE=1.E-5**             00000440
C       HKX   REAL VECTOR VARIABLE, STREAMWISE MESH                      00000450
```

```
C      HNY    REAL VECTOR VARIABLE, ETA MESH                               00000460
C      HNZ    REAL VECTOR VARIABLE, Z MESH                                 00000470
C     HMAX    REAL CONSTANT VARIABLE, MAXIMUM MESH-SIZE IN REFINEMENT OF   00000480
C             Y MESH,**INPUT, DEFAULT VALUE=2.1**                          00000490
C     HNEW    REAL VECTOR VARIABLE, USED IN VERTICAL MESH REFINEMENT       00000500
C             ROUTINE                                                      00000510
C        I    INTEGER CONSTANT VARIABLE, NUMBER OF POINTS IN Z-MESH (=THE  00000520
C             NUMBER OF INTERVALS + 1).  I MUST BE SUPPLIED BY USER IN     00000530
C             SUBROUTINE ZMESH IF ZSUPLY=.TRUE.,**INPUT, DEFAULT           00000540
C             VALUE=11 **                                                  00000550
C     IFMX    INTEGER CONSTANT VARIABLE, MAXIMUM MESH SUB-DIVISION IN      00000560
C             VERTICAL MESH REFINEMENT                                     00000570
C       IZ    INTEGER VARIABLE DENOTING THE Z-STATION BEING SOLVED         00000580
C        J    INTEGER CONSTANT VARIABLE, NUMBER OF VERTICAL POINTS,        00000590
C             (=THE NUMBER OF INTERNAL INTERVALS + 1), J MUST BE SUPPLIED  00000600
C             BY USER IN SUBROUTINE NAMED  YMESH IF YSUPLY=.TRUE..         00000610
C             **INPUT, DEFAULT VALUE=101**                                 00000620
C     JMAX    INTEGER CONSTANT VARIABLE, MAXIMUM NUMBER OF VERTICAL        00000630
C             POINTS ALLOWED                                               00000640
C    KSAME    INTEGER CONSTANT VARIABLE, USED IN VERTICAL MESH REFINE-     00000650
C             MENT, =0 IF THERE IS NO CHANGE IN DISTRIBUTION OF VERTICAL   00000660
C             MESH                                                         00000670
C     KTER    INTEGER CONSTANT VARIABLE, =0 IF NEWTON≠S METHOD FAILS TO    00000680
C             CONVERGE                                                     00000690
C     KPRT    INTEGER VECTOR VARIABLE, STREAMWISE STATIONS AT WHICH THE    00000700
C             SOLUTION IS TO BE PRINTED ON PAPER, STATEMENT MUST BE ADDED  00000710
C             TO SUBROUTINE PRMESH TO INDICATE THE STATIONS IF OPTPT=      00000720
C             .TRUE.. THE VECTOR MUST BE SUPPLIED IF XSUPLY=.TRUE.         00000730
C    NTEST    INTEGER CONSTANT VARIABLE, CRITERION IN CONVERGENCE TEST     00000740
C             OF NEWTON≠S METHOD                                           00000750
C   MAXITS    INTEGER CONSTANT VARIABLE, MAXIMUM NUMBER OF ITERATIONS      00000760
C             ALLOWED IN NEWTON≠S METHOD                                   00000770
C       NC    INTEGER VECTOR VARIABLE, ROW PERMUTATION VECTOR USED IN      00000780
C             SOLUTION OF BLOCK TRIDIAGONAL SYSTEM                         00000790
C      NCH    INTEGER VECTOR VARIABLE, PIVOTING STRATEGY INFORMATION       00000800
C             VECTOR                                                       00000810
C       NR    INTEGER VECTOR VARIABLE, COLUMN PERMUTATION VECTOR USED      00000820
C             IN SOLUTION OF BLOCK TRIDIAGONAL SYSTEM                      00000830
C       NU    INTEGER CONSTANT VARIABLE, NUMBER OF EQUATIONS (=6) TO BE    00000840
C             SOLVED                                                       00000850
C       NX    INTEGER CONSTANT VARIABLE, NUMBER OF STREAMWISE STATIONS TO  00000860
C             BE MARCHED, THIS MUST BE SUPPLIED IN SUBROUTINE XMESH IF     00000870
C             XSUPLY=.TRUE.. **INPUT, DEFAULT VALUE=60**                   00000880
C   NPRINT    INTEGER CONSTANT VARIABLE, NUMBER OF UNIFORM STREAMWISE      00000890
C             STATIONS AT WHICH SOLUTION IS TO BE PRINTED (XB-XA)/NPRINT,  00000900
C             **INPUT, DEFAULT VALUE=10**                                  00000910
C    OPTPT    LOGICAL VARIABLE, =.TRUE. IF USER IS TO SUPPLY STREAMWISE    00000920
C             STATIONS TO BE PRINTED ON PAPER, **INPUT, DEFAULT VALUE=     00000930
C             .FALSE.**                                                    00000940
C       PI    PARAMETERS IN THE PDE (I=1,2,....16)  SEE REFERENCE 2.       00000950
C     QINT    REAL VECTOR, ENTRAINMENT QUANTITY                            00000960
```

```
C       R   REAL CONSTANT VARIABLE,REYNOLDS NUMBER, **INPUT, DEFAULT     00000970
C           VALUE=1.0/14.21686ZE-5 **                                    00000980
C  REFINE   LOGICAL VARIABLE, =.TRUE. IF VERTICAL MESH IS TO BE REFINED,00000990
C           **INPUT, DEFAULT VALUE=.TRUE.**                              00001000
C     RNU   REAL CONSTANT VARIABLE, KINEMATIC VISCOSITY                  00001010
C  SEPRAT   LOGICAL VARIABLE, TAKES ON THE VALUE .TRUE. IF SEPARATION    00001020
C           OCCURS                                                       00001030
C   TAUWU   REAL VECTOR, SHEAR STRESS AT BODY                            00001040
C   TITLE   REAL VECTOR, CONTAINS TITLE OF CASE BEING SOLVED, **INPUT,   00001050
C           NO DEFAULT VALUE**                                           00001060
C   THETA   REAL VECTOR, STREAM LINE SLOPES                              00001070
C      UH   REAL VECTOR VARIABLE, USED IN SETUP OF BLOCK TRIDIAGONAL     00001080
C           SYSTEM                                                       00001090
C     UHX   REAL VECTOR VARIABLE, USED IN SETUP OF BLOCK TRIDIAGONAL     00001100
C           SYSTEM                                                       00001110
C    UMAX   REAL VECTOR, MAXIMUM VALUE OF U                              00001120
C  USTORE   REAL STORAGE MATRIX CONTAINING THE SOLUTION                  00001130
C  USUPLY   LOGICAL VARIABLE, =.TRUE. IF USER SUPPLIES HIS OWN INITIAL   00001140
C           CONDITIONS (VELOCITY PROFILES).  IF .FALSE., SIMILARITY      00001150
C           SOLUTIONS WILL BE GENERATED AND USED AS STARTING CONDITIONS 00001160
C           **INPUT, DEFAULT VALUE=.TRUE. **                             00001170
C      UT   REAL VECTOR VARIABLE, STEAMWISE VELOCITY COMPONENT AT THE    00001180
C           PRESENT STREAMWISE STATION, UT(K,L) HAS THE VALUE OF KTH     00001190
C           COMPONENT OF UT AT ETA = ETA(L), WHERE ETA(1)=YA=0 AND       00001200
C           ETA(M)=ETA(M-1)+H(M-1), M=2,3,...,J-1. THUS UT(1,3) HAS      00001210
C           VALUE OF F AT ETA(3)                                         00001220
C     UTX   REAL VECTOR VARIABLE, STREAMWISE VELOCITY COMPONENT AT THE   00001230
C           PREVIOUS STATION                                             00001240
C      XA   REAL CONSTANT VARIABLE, STARTING STREAMWISE STATION,         00001250
C           **INPUT, DEFAULT VALUE=0.0 **                                00001260
C      XB   REAL CONSTANT VARIABLE, THE RIGHT END-POINT OF STREAMWISE    00001270
C           MESH, (I.E. THE LAST STREAMWISE STATION),**INPUT, DEFAULT    00001280
C           VALUE=1.**                                                   00001290
C   XPLOT   REAL VECTOR, STREAMWISE STATIONS FOR FIRST TWO PLOTS         00001300
C     XPT   REAL CONSTANT VARIABLE, UNIFORM INTERVAL TO WHICH SOLUTION   00001310
C           IS TO BE PRINTED =(XB-XA)/NPRINT                             00001320
C  XSUPLY   LOGICAL VARIABLE, =.TRUE. IF USER SUPPLIES THE STREAMWISE    00001330
C           MESH,**INPUT, DEFAULT VALUE=.FALSE.**                        00001340
C    XTAU   REAL VECTOR, STREAMWISE STATIONS USED IN PLOTTING            00001350
C      YA   REAL CONSTANT VARIABLE, THE LEFT END-POINT OF VERTICAL       00001360
C           MESH, **INPUT, DEFAULT VALUE=0.0 **                          00001370
C      YB   REAL CONSTANT VARIABLE, THE RIGHT END-POINT OF VERTICAL      00001380
C           MESH,**INPUT, DEFAULT VALUE=20.**                            00001390
C    YBIG   REAL VECTOR, VALUE OF Y WHERE U ATTAINS ITS MAXIMUM          00001400
C  YSUPLY   LOIGCAL VARIABLE, =.TRUE. IF USER SUPPLIES THE VERTICAL      00001410
C           MESH,**INPUT, DEFAULT VALUE=.FALSE.**                        00001420
C      ZA   REAL CONSTANT VARIABLE, THE LEFT END-POINT OF Z-MESH,        00001430
C           **INPUT, DEFAULT VALUE=0.0 **                                00001440
C      ZB   REAL CONSTANT VARIABLE, THE RIGHT END-POINT OF Z-MESH,       00001450
C           **INPUT, DEFAULT VALUE=1.0 **                                00001460
C  ZSUPLY   LOGICAL VARIABLE, =.TRUE. IF USER SUPPLIES HIS OWN Z-MESH,   00001470
```

```
C          **INPUT, DEFAULT VALUE=.FALSE.**
C *                                                               *
C *   TROUBLE SHOOTING AND SOME HINTS IN USING THIS PROGRAM       *
C *                                                               *
C ***************************************************************
C
C IF OSCILLATIONS ARE DETECTED IN THE SOLUTION BETWEEN SUCCESSIVE
C POINTS IN EITHER STREAMWISE OR VERTICAL DIRECTIONS, MESH SHOULD BE
C HALVED IN THAT DIRECTION AND CALCULATIONS SHOULD BE DONE AGAIN TO
C CHECK IF THE OSCILLATION IS TRULY A PHYSICAL PHENOMENON, RATHER
C THAN NUMERICAL STABILITY. MESHES SHOULD BE SO CHOSEN TO ENSURE
C SOLUTIONS AGREE TO NUMERICAL ACCURACY WHEN MESHES ARE HALVED
C IF THE SOLUTION AT ETA=YB DOES NOT QUIET DOWN SUFFICIENTLY, YB SHOULD
C BE INCREASED
C TO ACHIEVE STRICTER CONVERGENCE IN NEWTON*S ITERATION, EPSERR SHOULD
C BE DECREASED (MUST BE GREATER THAN 1.E-13 WITH CDC 6600) AND MAXITS
C INCREASED. THESE CAN BE DONE BY CHANGING STATEMENTS IN THE MAIN
C PROGRAM.
C IF MORE ACCURATE SOLUTION IS DESIRED, RICHARDSON EXTRAPOLATION CAN
C BE USED TO GET HIGHER ORDER SOLUTION IN BOTH STREAMWISE AND VERTICAL
C DIRECTIONS.
C IF MORE POINTS ARE DESIRED, JMAX SHOULD BE INCREASED BY CHANGING THE
C STATEMENT IN THE MAIN PROGRAM. -THE COMMON STATEMENTS ALSO HAVE TO
C BE CHANGED ACCORDINGLY.
C
C
C REFERENCE 1  H.B. KELLER   -  A NEW DIFFERENCE SCHEME FOR PARABOLIC
C                               PROBLEMS, APPEARED IN  NUMERICAL
C                               SOLUTIONS OF PARTIAL DIFFERENTIAL
C                               EQUATIONS II, PP. 327-350, 1971.
C REFERENCE 2  MURPHY, W.D. -  A COMPUTATIONAL MODEL FOR THREE
C              SHANKAR, V.     DIMENSIONAL INCOMPRESSIBLE WALL JETS WITH
C              MALMUTH, N.D.   LARGE CROSS FLOW, NADC FINAL REPORT, 1978
C
C
C THIS FORTRAN IV SOURCE CODE WAS WRITTEN BY W. D. MURPHY, SCIENCE
C CENTER, ROCKWELL INTERNATIONAL, P.O. BOX 1085, THOUSAND OAKS, CA 91360
C AND WAS PREPARED FOR THE NAVAL AIR DEVELOPMENT CENTER, WARMINSTER,
C PA. 18974.
C
C
      LOGICAL REL,XSUPLY,YSUPLY,ZSUPLY,REFINE,OPTPT,USUPLY
      LOGICAL SEPRAT
      COMMON /SEPRAT/ SEPRAT
      COMMON /WRTE/ XPT,OPTPT,NPRINT
      COMMON /NET/ JMAX,HMAX,IFMX,KSAME
      COMMON /OPTION/ XSUPLY,YSUPLY,ZSUPLY,REFINE,USUPLY
      COMMON /MESHX/ HKX(401)/MESHY/ HKY(1 1) /MESHZ/ HKZ(51)
      COMMON /UT/ UT(6,101) /UTX/ UTX(6,101) /DU/ DU(6,101) /G/ G(6,101)
      COMMON /USTORE/ USTORE(6,101,11) /ULAST/ ULAST(6,101,11)
      COMMON /A/ A(6,6,101) /H/ H(4,6,101) /C/ C(2,6,101) 7F/ F(6,101)
      COMMON /NR/ NR(6,101) /NC/ NC(6,101) /NCK/ NCK(6,101)             00001990
      COMMON /XPRT/ XPRT(51)                                            00002000
      COMMON /HNEW/ HNEW(101)                                           00002010
      COMMON /PARM3/ MAXITS,EPSERR,KTER,REL,CUTOFF,MTEST                00002020
      COMMON /PARM4/ I,J,L/                                             00002030
      COMMON /PARM5/ FACX,HKS,XA,XB,YA,YB,ZA,ZB,NX                      00002040
      COMMON /KPPT/ KPPT(50)                                            00002050
      COMMON /RSQRT/ RSQRT                                              00002060
      COMMON /CCFACT/ FACTOR                                            00002070
      COMMON /CCPOOL/ XMIN,XMAX,YMIN,YMAX,CCXMIN,CCXMAX,CCYMIN,CCYMAX   00002080
      COMMON /TAUMU/ TAUMU(100,5) /XTAU/ XTAU(100)                      00002090
      COMMON /RNU/ RNU                                                  00002100
      COMMON /ETA/ ETA(101)                                             00002110
      COMMON /XPLOT/ XPLOT(5)                                           00002120
      COMMON /THETA/ THETA(65,11)                                       00002130
      COMMON /UMAX/ UMAX(65,5)                                          00002140
      COMMON /YBIG/ YBIG(65,5)                                          00002150
      COMMON /QINT/ QINT(65)                                            00002160
      DIMENSION LABEL1(2),LABEL2(2),LABEL3(2),LABEL4(2),LABEL5(2)       00002170
      DIMENSION LABEL6(2),LABEL7(2)                                     00002180
      DIMENSION TITLE(8)                                                00002190
      DIMENSION ISTOR(5)                                                00002200
      DIMENSION IP(65)                                                  00002210
      NAMELIST /INPUTS/ XSUPLY,YSUPLY,ZSUPLY,XA,XB,YA,YB,ZA,ZB,HKS,FACX,00002220
     X              NX,J,I,REFINE,HMAX,OPTPT,NPRINT,USUPLY,R            00002222
      DATA (ISTOR(L),L=1,5) /2,4,6,8,11/                                00002230
C                                                                       00002240
C    THE FOLLOWING CARDS ARE PLOT LABELS AND MAY BE CHANGED FOR EACH    00002250
C    CASE STUDY.                                                        00002260
      DATA LABEL1/10H  Y VS U/U,10HF  CASE 1  /                         00002270
      DATA LABEL2/10H  Y VS W/W,10HF  CASE 7  /                         00002280
      DATA LABEL3/10HSHEAR STRE,10HSS VS X    /                         00002290
      DATA LABEL4/10HSTREAM LIN,10HES CASE 1  /                         00002300
      DATA LABEL5/10HUMAX VS X ,10H CASE 7    /                         00002310
      DATA LABEL6/10HJET SPREAD,10HING VS X   /                         00002320
      DATA LABEL7/10HENTRAINMEN,10HT QUANTITY/                          00002330
      DATA IP(1),IP(2),IP(3),IP(4),IP(5) /10,20,30,35,40/               00002340
C                                                                       00002350
C    DEFAULT VALUES                                                     00002360
      OPTPT=.FALSE.                                                     00002370
      XSUPLY=.FALSE.                                                    00002380
      YSUPLY=.FALSE.                                                    00002390
      ZSUPLY=.FALSE.                                                    00002400
      REFINE=.TRUE. .                                                   00002410
      USUPLY=.TRUE.                                                     00002420
      NPRINT=10                                                         00002430
      FACX=1.2                                                          00002440
      HKS=1.E-5                                                         00002450
      XA=1.0                                                            00002460
      YA=0.0                                                            00002470
      ZA=0.0                                                            00002480
      XH=1.
      YH=20.
      ZH=1.0
      J=101
      I=11
      NX=60
      HMAX=2.1
      APT=XH/NPRINT
      KC=1
      R=1.0/14.216862E-5
      RNU=14.216862E-5
C
C     INITIALIZE
C
      JMAX=101
      MAXITS=50
      EPSERR=1.0E-2
      REL=.FALSE.
      CUTOFF=1.0E-3
      NTEST=5
      KSTART=1
      KTEN=1
      DO 5 L=1,65
    5 QINT(L)=0.0
      DO 30 L=1,JMAX
      DO 20 K=1,6
      G(K,L)=1.
      UTX(K,L)=0.
   20 CONTINUE
   30 CONTINUE
      READ(5,35) TITLE
   35 FORMAT(8A10)
      WRITE(6,36) TITLE
   36 FORMAT(1H1,10X,8A10,//)
      READ(5,INPUTS)
      RSQRT=SQRT(R)
      CALL PREP(KSTART)
      WRITE(6,INPUTS)
      CCXMIN=100.
      CCYMIN=100.
      NX1=10
      CCXMAX=600.
      CCYMAX=600.
      CCX=.5*(CCXMAX-CCXMIN)-50.
      CCY=650.
      NY1=10
      FACTOR=1.0
      XMIN=0.0
      YMIN=0.0
      XMAX=2.0
      YMAX=.5
      CALL WALJFT(XSTART)
      CALL PROFLE
C
C   REMOVE ALL OF THE NEXT CARDS EXCEPT THE LAST TWO IF THIS PROGRAM IS
C   NOT BEING RUN ON THE BERKELEY CDC 7600 OR IF PLOTS ARE NOT DESIRED.
C
      JJ=J
      PROD2=1.6
      IF(I.EQ.1) GO TO 1010
      IF(SEPRAT) GO TO 210
      DO 200 K=1,5
      IT=ITSTOR(K)
      CALL CCNEXT
      CALL CCGRID(NX1,2,6HLABELS,NY1,2)
      CALL CCLTR(CCX,CCY,0,2,LABEL1,20)
      DO 100 IP=1,3
      PROD1=PROD2
C ***** THE NEXT UE CARD MUST BE CHANGED FOR EACH CASE STUDY *******
C *                                                               *
      UE=1.0/SQRT(XPLOT(IP))
C *                                                               *
C *****************************************************************
      PROD2=SQRT(XPLOT(IP)*RNU/UE)
      L1=1
      DO 95 L=1,JJ
      ETA(L)=PROD2*ETA(L)/PROD1
      IF(ETA(L).GT.YMAX.AND.ETA(L1).LE.YMAX) J=L1
      L1=L
      HNEW(L)=ULAST(IP,L,IT)
      IF(HNEW(L).GT.XMAX) HNEW(L)=XMAX
   95 CONTINUE
  100 CALL CCPLOT(HNEW,ETA,J,4HJOIN)
      CALL CCNEXT
      CALL CCGRID(NX1,2,6HLABELS,NY1,2)
      CALL CCLTR(CCX,CCY,0,2,LABEL2,20)
      DO 120 IP=4,6
      PROD1=PROD2
C ***** THE NEXT UE CARD MUST BE CHANGED FOR EACH CASE STUDY *******
C *                                                               *
      UE=1.0/SQRT(XPLOT(IP-3))
C *                                                               *
C *****************************************************************
      PROD2=SQRT(XPLOT(IP-3)*RNU/UE)
      L1=1
      DO 110 L=1,JJ
      ETA(L)=PROD2*ETA(L)/PROD1
      IF(ETA(L).GT.YMAX.AND.ETA(L1).LE.YMAX) J=L1
      HNEW(L)=ULAST(IP,L,IT)
      IF(HNEW(L).GT.XMAX) HNEW(L)=XMAX
      L1=L
  110 CONTINUE
```

```
120 CALL CCPLOT([illegible]A,J,4HJOIN)                                00003510
200 CONTINUE                                                          00003520
210 CONTINUE                                                          00003530
    NX=NX-1                                                           00003540
    XMIN=XA                                                           00003550
    XMAX=XB                                                           00003560
    YMAX=1.0E3                                                        00003570
    CALL CCNEXT                                                       00003580
    CALL CCGRID(NX1,2,6HLABELS,NY1,2)                                 00003590
    CALL CCLTR(CCX,CCY,0,2,LABEL3,20)                                 00003600
    DO 250 K=1,5                                                      00003610
250 CALL CCPLOT(XTAU(2),TAUMU(2,K),NX,4HJOIN)                         00003620
    YMAX=2.0                                                          00003630
    CALL CCNEXT                                                       00003640
    CALL CCGRID(NX1,2,6HLABELS,NY1,2)                                 00003650
    CALL CCLTR(CCX,CCY,0,2,LABEL5,20)                                 00003660
    DO 260 K=1,5                                                      00003670
260 CALL CCPLOT(XTAU(2),UMAX(2,K),NX,4HJOIN)                          00003680
    YMAX=.2                                                           00003690
    CALL CCNEXT                                                       00003700
    CALL CCGRID(NX1,2,6HLABELS,NY1,2)                                 00003710
    CALL CCLTR(CCX,CCY,0,2,LABEL6,20)                                 00003720
    DO 270 K=1,5                                                      00003730
270 CALL CCPLOT(XTAU(2),YHIG(2,K),NX,4HJOIN)                          00003740
    YMAX=0.0                                                          00003750
    DO 280 L=2,NX                                                     00003760
    IF(YMAX.GE.QINT(L)) GO TO 280                                     00003770
    YMAX=QINT(L)                                                      00003780
280 CONTINUE                                                          00003790
    YMAX=YMAX+.001                                                    00003800
    CALL CCNEXT                                                       00003810
    CALL CCGRID(NX1,2,6HLABELS,NY1,2)                                 00003820
    CALL CCLTR(CCX,CCY,0,2,LABEL7,20)                                 00003830
    CALL CCPLOT(XTAU(2),QINT(2),NX,4HJOIN)                            00003840
    L=5                                                               00003850
    IF(NX.LT.42) GO TO 505                                            00003860
    DO 500 LL=42,NX                                                   00003870
    L=L+1                                                             00003880
500 IP(L)=LL                                                          00003890
505 CONTINUE                                                          00003900
    XMIN=XA                                                           00003910
    YMAX=ZH                                                           00003920
    CCXMAX=500.                                                       00003930
    NX1=4                                                             00003940
    NY1=5                                                             00003950
    CCX=.5*(CCXMAX-CCXMIN)-50.                                        00003960
    CALL CCNEXT                                                       00003970
    CALL CCGRID(NX1,2,6HLABELS,NY1,2)                                 00003980
    CALL CCLTR(CCX,CCY,0,2,LABEL4,20)                                 00003990
    I1=I-1                                                            00004000
    LSUM=1                                                            00004010
```

```
      X=XA                                                              00004020
      DO 600 L=2,NX                                                     00004030
      X=X+HKX(L)                                                        00004040
      IF(L.NE.IP(LSUB)) GO TO 600                                       00004050
      HNEW(1)=X                                                         00004060
      LSUB=LSUB+1                                                       00004070
      Z=ZA                                                              00004080
      DO 550 I=1,II                                                     00004090
      ETA(1)=Z                                                          00004100
      TH=THETA(L,I)                                                     00004110
      CTH=COS(TH)                                                       00004120
      STH=SIN(TH)                                                       00004130
      HNEW(2)=.25*CTH+X                                                 00004140
      ETA(2)=Z+.25*STH                                                  00004150
      IF(HNEW(1).GT.XMAX.OR.ETA(1).GT.ZB) GO TO 550                     00004160
      IF(HNEW(2).GT.XMAX.OR.ETA(2).GT.ZB) GO TO 550                     00004170
  540 CALL CCPLOT(HNEW,ETA,2,4HJOIN)                                    00004180
  550 Z=Z+HKZ(I)                                                        00004190
  600 CONTINUE                                                          00004200
 1000 CALL CCEND                                                        00004210
 1010 STOP                                                              00004220
      END                                                               00004230
      SUBROUTINE NEWTON(KASE,N1,N2,N3)                                  00004240
C                                                                       00004250
C   THIS SUBROUTINE SOLVES THE 2-D ATTACHMENT LINE EQUATION WHEN KASE=2 00004260
C   AND THE 3-D WALL JET EQUATION WHEN KASE=3.  NEWTON ITERATIONS ARE   00004270
C   EMPLOYED.                                                           00004280
C                                                                       00004290
      LOGICAL REL                                                       00004300
      COMMON /THETA/ THETA(65,1)                                        00004310
      COMMON /G/ G(6,1)                                                 00004320
      COMMON /PARM1/ N,NP,NO                                            00004330
      COMMON /PARM3/ MAXITS,EPSERR,KTER,REL,CUTOFF,MTEST                00004340
      COMMON /PARM4/ I,J,IZ                                             00004350
      COMMON /PARM6/ KIUNIT,XN,HX,X,ZN,HZ,Z                             00004360
      COMMON /UT/ UI(6,1) /UU/ BU(6,1)                                  00004370
      COMMON /TAUMU/ TAUMU(100,5) /XTAU/ XTAU(100)                      00004380
      COMMON /GINT/ OINT(65)                                            00004390
      COMMON /MESHY/ HKY(1) /MESHZ/ HKZ(1)                              00004400
      COMMON /KNU/ KNU                                                  00004410
      COMMON /UMAX/ UMAX(65,5)                                          00004420
      COMMON /YHIG/ YHIG(65,5)                                          00004430
      COMMON /PREPMP/ IFLAG,YTC,YTOLD,YTOLD2,ITER,YTCC                  00004440
      EXTERNAL WHSF,HC,WHSF2D                                           00004450
      DATA OMEGA/.7/                                                    00004460
      DATA PI/3.14159265358979/                                         00004470
C                                                                       00004480
C INITIALIZE                                                            00004490
C                                                                       00004500
      N=N1                                                              00004510
      NP=N2                                                             00004520
      NO=N3
      IFLAG=1
      YTC=0.
      YTOLD=0.
C
C CONTRIBUTION FROM PREVIOUS STATIONS
C
      IF(KASE.EQ.3) GO TO 10
      CALL PREPG2
      IF(KOUNT.EQ.1) GO TO 15
      CALL PREPP(KN,7,1)
      GO TO 15
   10 CALL PREPG
   15 CONTINUE
C
C NEWTON ITERATION
C
      DO 300 M=1,MAXITS
      ITER=M
C
C   SOLVE 2-D PROBLEM
C
      IF(KOUNT.GT.1) CALL PREPD
      IF(KASE.EQ.2) CALL BOX(KASE,RHSF2D,BC)
C
C   SOLVE 3-D PROBLEM
C
      IF(KASE.EQ.3) CALL BOX(KASE,RHSF,BC)
C
C CALL BLOCK TRIDIAGONAL MATRIX SOLVER
C
      CALL BLOCK1
      CALL BLOCK2
C
C
C ERROR TESTING BETWEEEN TWO NEWTON ITERATES
C
      ERMAX=0.
      DO 200 L=1,J
C
C UNDERRELAXATION
C
      DO 200 K=1,N
      UT(K,L)=UT(K,L)+OMEGA*DU(K,L)
      ERROR=ABS(DU(K,L))
C
C ABSOLUTE ERROR CRITERION IS USED IF VARIABLE ** REL ** IS SET TO
C .FALSE.
C
      IF(.NOT.REL) GO TO 195
      IF(ABS(UT(K,L)).GT.1.E-4) ERROR=ERROR/ABS(UT(K,L)+0.5*DU(K,L))
```

```
  195 CONTINUE
      IF(ERMAX.GE.ERROR) GO TO 200
      LOCX=K
      LOCY=L
      ERMAX=ERROR
  200 CONTINUE
C
      IF(ERMAX.GE.EPSERR) GO TO 205
      WRITE(6,6400) M,ERMAX,LOCX,LOCY,YICC,UT(3,1),UT(6,1)
C
C COMPUTE PLOT DATA
C
C **** THE NEXT UE AND WE CARDS MUST BE CHANGED FOR EACH CASE STUDY ****
C *
      UE=1.0/SQRT(XN)
      WE=1.0E-5*UE*ZN
C *
C ****************************************************************************
      TERM=SQRT(UE*RMU*XN)
      J1=J-1
      SUM=0.0
      DO 700 L=1,J1
  700 SUM=SUM+.5*HKY(L)*(UT(2,L)+UT(2,L+1))
      SUM=TERM*SUM
      IF(IZ.EQ.1) GO TO 710
      QINT(KOUNT)=QINT(KOUNT)+.5*HKZ(IZ-1)*(SUM+SUM1)
  710 SUM1=SUM
      ARG=WE*UT(6,1)/(UE*UT(3,1))
      THETA(KOUNT,IZ)=ATAN(ARG)
      PROD=UE/(RMU*XN)
      Q=SQRT(PROD*((UE*UT(3,1))**2+(WE*UT(6,1))**2))
      IF(IZ.NE.1) GO TO 201
      TAUMU(KOUNT,1)=Q
      XTAU(KOUNT)=XN
      CALL VELMAX(UMAX(KOUNT,1),YBIG(KOUNT,1),UE)
      GO TO 750
  201 IF(IZ.NE.3) GO TO 202
      TAUMU(KOUNT,2)=Q
      CALL VELMAX(UMAX(KOUNT,2),YBIG(KOUNT,2),UE)
      GO TO 750
  202 IF(IZ.NE.6) GO TO 203
      TAUMU(KOUNT,3)=Q
      CALL VELMAX(UMAX(KOUNT,3),YBIG(KOUNT,3),UE)
      GO TO 750
  203 IF(IZ.NE.8) GO TO 204
      TAUMU(KOUNT,4)=Q
      CALL VELMAX(UMAX(KOUNT,4),YBIG(KOUNT,4),UE)
      GO TO 750
  204 IF(IZ.NE.10) GO TO 750
      TAUMU(KOUNT,5)=Q
      CALL VELMAX(UMAX(KOUNT,5),YBIG(KOUNT,5),UE)
  750 CONTINUE
      WRITE(6,6200) G(2,10),G(2,20),G(2,30),G(2,40),G(2,50),G(2,J),Q
 6200 FORMAT(1H ,5X,*H=*,6E13.5,2X,*TAU=*,E13.5)
      IF(IZ.EQ.1) WRITE(6,6500) QINT(KOUNT),(UMAX(KOUNT,L),L=1,5),
     X(YBIG(KOUNT,K),K=1,5)
 6500 FORMAT(1H ,2X,*U=*,E13.5,2X,*UMAX=*,5E13.5,//,19X,*YMAX=*,5E13.5//
     X)
      RETURN
  205 CONTINUE
      IF(M.LT.MTEST) GO TO 210
      IF((ERMAX.LT.CUTOFF).AND.(EROLD/ERMAX.LT.1.05)) RETURN
  210 CONTINUE
      EROLD=ERMAX
C
  300 CONTINUE
C
C NO CONVERGENCE, WRITE FINAL ITERATE AND EXIT
C
      KIER=0
      WRITE(6,6100)
C
 6000 FORMAT(* NEWTON - ITER =*,I2,* ERROR =*,E12.5,* AT(*,I2,*,*,I3,
     1   * )*,5X,* YTC = *,F12.5,* U(3,1) *,E12.5,* T *,E12.5)
 6100 FORMAT(* NEWTON - NO CONVERGENCE IN NEWTON ITERATION *)
C
      RETURN
      END
      SUBROUTINE PREPF(X,Z,ITYPE)
C
C   THIS SUBROUTINE COMPUTES THE VALUES OF P1,P2,...,P10, WHICH ARE
C   FUNCTIONS OF UE,WE,K1,K2,H1, AND H2.
C
      REAL K1,K2
      COMMON /PARM2/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10
      COMMON /PARM4/ I,J,IZ
      COMMON /PARM6/ KOUNT,XM,HX,XX,ZN,HZ,ZZ
      COMMON /UFWE/ UF,WF
      COMMON /K1K2/ K1,K2
      DATA PI/3.14159265358979/
      K1=0.0
      K2=0.0
C **** NEXT CARDS FOR UE,UEX,UEZ,WE,WEX,WEZ, AND WEZX DEPEND UPON CASE *
C *
      XROOT=SQRT(X)
      UE=1.0/XROOT
      UEX=-.5/(X*XROOT)
      UEZ=0.0
      WE=1.0E-5*UE*Z
      WEX=1.0E-5*UEX*Z
      WEZ=1.0E-5*UE
      WEZX=1.0E-5*UEX
```

```
C *
C ***********************************************************************
      P2=X*UEX/UE
      P1=.5*(1.0+P2)
      P3=X*WEZ/UE
      P5=WE*X*UEZ/(UE*UE)
      P6=P3-.5*P5
      P7=WE/UE
      P8=6.0
      P9=0.0
      P10=XX
      IF(IZ.EQ.1) GO TO 10
      P4=X*WEX/WE
      RETURN
C
C   ATTACHMENT LINE PARAMETERS
C
   10 P4= X*WEZZ/WEZ
      RETURN
      END
      SUBROUTINE VELMAX(BIG,YBIG,UE)
C
C   THIS ROUTINE COMPUTES THE MAXIMUM VALUE OF U AND THE CORRESPONDING
C   VALUE OF Y.
C
      COMMON /PARM4/ I,J,IZ
      COMMON /PARM6/ KOUNT,XN,HX,X,ZN,HZ,Z
      COMMON /UT/ UT(6,1)
      COMMON /ETA/ ETA(1)
      COMMON /RNU/ RNU
      BIG=0.0
      DO 20 L=1,J
      TEST=UE*UT(2,L)
      IF(BIG.GE.TEST) GO TO 20
      BIG=TEST
      IBIG=L
   20 CONTINUE
      IF(BIG.GT.2.0) BIG=2.0
      YBIG=ETA(IBIG)*SQRT(RNU*XN/UE)
      RETURN
      END
      SUBROUTINE PREP(KSTART)
C
C   THIS ROUTINE SETS UP THE MESHES OR CALLS MESH SUBROUTINES.  EITHER
C   INITIAL PROFILES ARE COMPUTED (USUPLY=.FALSE.) FROM SIMILARITY
C   SOLUTIONS OR THEY ARE SUPPLIED BY THE USER.
C
      LOGICAL REL,XSUPLY,YSUPLY,ZSUPLY,REFINE,OPTPT,USUPLY
      COMMON /OPTION/ XSUPLY,YSUPLY,ZSUPLY,REFINE,USUPLY
      COMMON /WRTE/ API,OPTPT,NPRINT
      COMMON /MESHX/ HX(1) /MESHY/ H(1)
      COMMON /MESH2/ HKZ(1)
      COMMON /PARM3/ MAXITS,EPSERR,KIER,REL,CUTOFF,MTEST
      COMMON /PARM4/ I,J,IZ
      COMMON /PARM5/ FACX,HKS,XA,XB,YA,YB,ZA,ZB,NX
      COMMON /PARM6/ KOUNT,XN,HX,X,/N,HZ,Z
      COMMON /NET/ JMAX,HMAX,IFMX,KSAME
      COMMON /UT/ UT(6,1) /UTX/ UTX(6,1)
      COMMON /USTORE/ USTORE(6,101,1)
      COMMON /RNU/ RNU
C
      F1(T)=(1.-EXP(-T))/(1.+EXP(-T))
      F2(T)=(1.-EXP(-T))/(1.+EXP(-T))
      F3(T)=2.*EXP(-T)/(1.+EXP(-T))**2
      N=6
      NP=4
      NG=2
C
C STREAMWISE MESH
C
      IF(.NOT.XSUPLY) GO TO 90
      CALL XMESH
      GO TO 150
   90 CONTINUE
      HKX(1)=HKS
      KT=1
      KL=1
      XT=-HKS
C
  100 CONTINUE
      IF((KL.GT.100).OR.(XT.GT.XB)) GO TO 140
      XT=XT+HKX(KL)
      IF(HKX(KL).GT.XPT) FACX=1.
      IF(XT.GE.(KT*XPT)) GO TO 110
      HKX(KL+1)=HKX(KL)*FACX
      KL=KL+1
      GO TO 100
  110 CONTINUE
      IF(FACX.EQ.1.) GO TO 130
      IF((XT-KT*XPT).GT.(HKX(KL)/20.)) GO TO 120
      HKX(KL+1)=HKX(KL)*FACX+(XT-KT*XPT)
      HKX(KL)=KT*XPT-(XT-HKX(KL))
      XT=KT*XPT
      KL=KL+1
      KT=KT+1
      GO TO 100
  120 CONTINUE
      HKX(KL+2)=HKX(KL)*FACX
      HKX(KL+1)=XT-KT*XPT
      HKX(KL)=KT*XPT-(XT-HKX(KL))
      KL=KL+2
      KT=KT+1
      GO TO 100                                                         00007050
  130 CONTINUE                                                          00007060
      HKX(KL)=XT*XPT-(XT-HKX(KL))                                       00007070
      XT=KT*XPT                                                         00007080
  135 CONTINUE                                                          00007090
      HKX(KL+1)=XPT                                                     00007100
      KT=KT+1                                                           00007110
      KL=KL+1                                                           00007120
      XT=XT+HKX(KL)                                                     00007130
      IF(XT.LE.XB) GO TO 135                                            00007140
  140 CONTINUE                                                          00007150
C                                                                       00007160
      NX=KL-1                                                           00007170
  150 CONTINUE                                                          00007180
C                                                                       00007190
      IF(OPTPT) CALL PRMESH                                             00007200
C                                                                       00007210
C VERTICAL MESH                                                         00007220
C                                                                       00007230
      IF(.NOT.YSUPLY) GO TO 350                                         00007240
      CALL YMESH                                                        00007250
      GO TO 400                                                         00007260
  350 CONTINUE                                                          00007270
      J1=J-1                                                            00007280
      JA=(J-1)*3./5.                                                    00007290
      JB=JA+1                                                           00007300
      JC=J-1-JA                                                         00007310
      HT=YB/3./JA                                                       00007320
      DO 160 L=1,JA                                                     00007330
      H(L)=HT                                                           00007340
  160 CONTINUE                                                          00007350
      HT=2*YB/3./JC                                                     00007360
      DO 170 L=JB,J1                                                    00007370
      H(L)=HT                                                           00007380
  170 CONTINUE                                                          00007390
  400 CONTINUE                                                          00007400
      IF(.NOT.ZSUPLY) GO TO 405                                         00007410
      CALL ZMESH                                                        00007420
      GO TO 420                                                         00007430
  405 DO 410 L=2,I                                                      00007440
  410 HKZ(L-1)=.1                                                       00007450
  420 CONTINUE                                                          00007460
C                                                                       00007470
C OBTAIN INITIAL PROFILES                                               00007480
C                                                                       00007490
      IF(USUPLY)CALL PROFLE                                             00007500
      XPT=XA                                                            00007510
      YPT=YA                                                            00007520
      ZPT=ZA                                                            00007530
C ***** THE NEXT DE CARD MUST BE CHANGED FOR EACH CASE STUDY *******    00007540
C *                                                                *    00007550
```

```
      UE=1.0
C *                                                                   *
C *********************************************************************
      PROD=SQRT(XA*RNU/UF)
      YYPT=PROD*YPT
      LMAX=MAX0(I,J,NX)
      WRITE(6,5800)
      L=1
C
C PRINT MESH POINTS
C
      WRITE(6,5900) L,XPT,YPT,ZPT,YYPT
      DO 450 L=2,LMAX
      LM1=L-1
      XOUT=0.0
      YOUT=0.0
      ZOUT=0.0
      YYOUT=0.0
      IF(L.GT.NX) GO TO 430
      XPT=XPT+HKX(L)
      XOUT=XPT
  430 IF(L.GT.J) GO TO 440
      YPT=YPT+H(LM1)
      YOUT=YPT
      YYOUT=YOUT*PROD
  440 IF(L.GT.I) GO TO 445
      ZPT=ZPT+HKZ(LM1)
      ZOUT=ZPT
  445 WRITE(6,5900) L,XOUT,YOUT,ZOUT,YYOUT
  450 CONTINUE
      IF(USUPLY) RETURN
C
      J1=J-1
C
C SOLVES FOR INITIAL SIMILAR SOLUTION
C
      IF(KSTART.GT.1) RETURN
      Y=YA
      H(J)=0.
      DO 195 L=1,J
      DO 190 K=1,N
      UTX(K,L)=0.
  190 CONTINUE
      DO 200 L=1,J
      UI(1,L)=F1(Y)
      UI(2,L)=F2(Y)
      UI(3,L)=F3(Y)
      UI(4,L)=UI(1,L)
      UI(5,L)=UI(2,L)
      UI(6,L)=UI(3,L)
      Y=Y+H(L)
  200 CONTINUE
C
C DUMMY INITIALIZATION
C
      AN=1.
      HX=1.
      KOUNT=1
      X=0.
      Z=0.0
      IZ=1
C
      CALL NEWTON(2,N,NP,NQ)
      IF(.NOT.REFINE) GO TO 2100
      CALL NETRUN
      IF(KSAME.EQ.0) WRITE(6,6000)
C
C RECOMPUTE SOLUTION ON THE REFINED NET
C
      IF(KSAME.EQ.1) CALL NEWTON(2,N,NP,NQ)
 2100 CONTINUE
      DO 2150 IZZ=1,I
      DO 2150 L=1,J
      DO 2150 K=1,N
      USTORE(K,L,IZZ)=UT(K,L)
 2150 CONTINUE
      WRITE(6,6100) KOUNT,X,X
      CALL OUTPT
      IF(KTER.EQ.0) STOP
C
 5800 FORMAT(1H1,16X,*GRID POINTS*/1H0,5X,*I*,6X,*X*,11X,*ETA*,10X,*Z*,
     X11X,*Y*//)
 5900 FORMAT(1H ,3X,I3,4(3X,F9.5))
 6000 FORMAT(* PREP - NO CHANGE IN REFINEMENT*)
 6100 FORMAT(* STATION = *,I4,*, X =*,E12.5,*, X-STEP =*,E12.5)
C
      RETURN
      END
      SUBROUTINE PREPM
C
C   THIS SUBROUTINE MODELS TURBULENCE USING TWO LAYERS AND DEPENDS UPON
C   WALL CURVATURE AND THE REYNOLDS NUMBER.  K1 AND K2 ARE THE GEODESIC
C   CURVATURES OF THE CURVES Z=CONST AND X=CONST,RESPECTIVELY.
C
      REAL K1,K2
      COMMON /USTORE/ US(6,101,1) /UTX/ UX(6,1)
      COMMON /HSQRT/ HSQRT
      COMMON /UEWE/ UE,WF
      COMMON /UT/ U(6,1) /G/ G(6,1) /MESHY/ H(1)
      COMMON /PARM4/ I,J,IZ
      COMMON /PARM5/ FACX,HKS,XA,XB,YA,YB,ZA,ZB,NX
      COMMON /PARM6/ KOUNT,XN,HX,X,ZN,HZ,Z
      COMMON /PREPBP/ IFLAG,YTC,YTOLD,YTOLD2,ITER,YTCC
      COMMON /KIK2/ K1,K2
C
C   TC1 AND TC2 CAN BE ASSIGNED VALUES BETWEEN 0 AND 3 BY CHANGING THE
C   NEXT DATA STATEMENT
      DATA TC1,TC2/0.0,0.0/
C
C U HAS THE SAME MEANING AS UT IN THE MAIN PROGRAM
C
      IZM1=IZ-1
      TERM=SQRT(1.0+(WE/UE)**2)
      S=X
      R=UE*S/14.716862E-5
      RSQRT=SQRT(R)
      IF(ITER.LT.5) GO TO 90
      IF(IFLAG.EQ.0) GO TO 400
      IF(((YTC-YTOLD).EQ.(-(YTOLD-YTOLD2))).AND.(YTOLD.NE.YTC)) IFLAG=0
      IF(IFLAG.EQ.1) GO TO 90
      YTC=AMAX1(YTC,YTOLD)
      YT=.435*YTC/.125
      GO TO 400
   90 CONTINUE
      YTOLD2=YTOLD
      YTOLD=YTC
      YT=YH
      J1=J-1
C
C DETERMINE POINT WHERE SECOND LAYER STARTS
C
      IF(IZ.EQ.1) GO TO 99
      DO 95 LL=1,J1
      L=J-LL
      YT=YT-H(L)
      FP=.25*(U(2,L)+UX(2,L)+US(2,L,IZ)+US(2,L,IZM1))
      GP=.25*(U(5,L)+UX(5,L)+US(5,L,IZ)+US(5,L,IZM1))
      UNORM=ABS(TERM-SQRT(FP**2+(WE*GP/UE)**2))/TERM
      IF(UNORM.LT.0.01) GO TO 95
      GO TO 200
   95 CONTINUE
      GO TO 200
   99 CONTINUE
      DO 100 LL=1,J1
      L=J-LL
      YT=YT-H(L)
      FP=.5*(U(2,L)+UX(2,L))
      GP=.5*(U(5,L)+UX(5,L))
      UNORM=ABS(TERM-SQRT(FP**2+(WE*GP/UE)**2))/TERM
      IF(UNORM.LT.0.01) GO TO 100
      GO TO 200
  100 CONTINUE
  200 CONTINUE
```

```
      YTC=YT*.125/.435
  400 CONTINUE
      YTCC=YTC
      PROD0=.125*RSQRT*YT*YT*.125
C
      Y=0.
      IF(IZ.NE.1) GO TO 450
      DO 300 L=1,J
      FP=.5*(U(2,L)+UX(2,L))
      GP=.5*(U(5,L)+UX(5,L))
      FPP=.5*(U(3,L)+UX(3,L))
      GPP=.5*(U(6,L)+UX(6,L))
      PROD1=SQRT(FPP*FPP+(WF*GPP/UE)**2)
      PROD2=1.0-TC1*FP*K2*S/(FPP*(RSQRT+K2*S*Y))
      PROD3=1.0
      PROD4=.435*RSQRT*Y*Y*.435
      IF(Y.GE.YTC) GO TO 250
C
C FIRST LAYER
C
      G(2,L)=1.0+PROD4*PROD1*PROD2
      G(6,L)=1.0+PROD4*PROD1*PROD3
      GO TO 290
C
C SECOND LAYER
C
  250 CONTINUE
      PROD2=1.0
      G(2,L)=1.0+PROD0*PROD1*PROD2
      G(6,L)=1.0+PROD0*PROD1*PROD3
  290 Y=Y+H(L)
  300 CONTINUE
      DO 350 L=2,J
      L1=L-1
      G(3,L)=G(3,L)+(G(2,L)-G(1,L))*UX(3,L)-(G(2,L1)-G(1,L1))*UX(3,L1)
      G(4,L)=G(4,L)+(G(6,L)-G(5,L))*UX(6,L)-(G(6,L1)-G(5,L1))*UX(6,L1)
  350 CONTINUE
      GO TO 510
  450 DO 460 L=1,J
      FP=.25*(U(2,L)+UX(2,L)+US(2,L,IZ)+US(2,L,IZM1))
      FPP=.25*(U(3,L)+UX(3,L)+US(3,L,IZ)+US(3,L,IZM1))
      GP=.25*(U(5,L)+UX(5,L)+US(5,L,IZ)+US(5,L,IZM1))
      GPP=.25*(U(6,L)+UX(6,L)+US(6,L,IZ)+US(6,L,IZM1))
      PROD1=SQRT(FPP*FPP+(WF*GPP/UE)**2)
      PROD2=1.0-TC1*FP*K2*S/(FPP*(RSQRT+K2*S*Y))
      PROD3=1.0-TC2*GP*K1*S/(GPP*(RSQRT+K1*S*Y))
      PROD4=.435*RSQRT*Y*Y*.435
      IF(Y.GE.YTC) GO TO 475
C
C FIRST LAYER
C
```

```
      G(2,L)=1.0+PROD4*PROD1*PROD2                                      00009590
      G(6,L)=1.0+PROD4*PROD1*PROD3                                      00009600
      GO TO 490                                                         00009610
C                                                                       00009620
C SECOND LAYER                                                         00009630
C                                                                       00009640
  475 CONTINUE                                                          00009650
      PROD2=1.0                                                         00009660
      PROD3=1.0                                                         00009670
      G(2,L)=1.0+PROD0*PROD1*PROD2                                      00009680
      G(6,L)=1.0+PROD0*PROD1*PROD3                                      00009690
  490 Y=Y+H(L)                                                          00009700
  400 CONTINUE                                                          00009710
      DO 500 L=2,J                                                      00009720
      L1=L-1                                                            00009730
      G(3,L)=G(3,L)+(G(2,L)-G(1,L))*(UX(3,L)+US(3,L,IZ)+US(3,L,IZM1))   00009740
     X        -(G(2,L1)-G(1,L1))*(UX(3,L1)+US(3,L1,IZM1)+US(3,L1,IZ))   00009742
      G(4,L)=G(4,L)+(G(6,L)-G(5,L))*(UX(6,L)+US(6,L,IZ)+US(6,L,IZM1))-  00009750
     X        (G(6,L1)-G(5,L1))*(UX(6,L1)+US(6,L1,IZ)+US(6,L1,IZM1))    00009752
  500 CONTINUE                                                          00009760
  510 CONTINUE                                                          00009770
      DO 550 L=1,J                                                      00009780
      G(1,L)=G(2,L)                                                     00009790
      G(5,L)=G(6,L)                                                     00009800
  550 CONTINUE                                                          00009810
      RETURN                                                            00009820
      END                                                               00009830
      SUBROUTINE OUTPT                                                  00009840
C                                                                       00009850
C   THIS SUBROUTINE WRITES THE SOLUTION ON PAPER FOR THE PLANE X=XN.    00009860
C                                                                       00009870
      COMMON /ULAST/ ULAST(6,10,1)                                      00009880
      COMMON /USTORE/ U(6,10,1)                                         00009890
      COMMON /PARM4/ I,J,IZ                                             00009900
      COMMON /PARM5/ FACX,MXS,XA,XB,YA,YB,ZA,ZB,NX                      00009910
      COMMON /PARM6/ KOUNT,XN,HX,X,ZN,HZ,Z                              00009920
      COMMON /XPLOT/ XPLOT(3)                                           00009930
      DIMENSION ISTOR(5)                                                00009940
      DATA (ISTOR(L),L=1,5) /1,4,6,8,11/                                00009950
      DATA ISTART/0/                                                    00009960
      IF(ISTART.NE.0.AND.KOUNT.EQ.1) GO TO 310                          00009970
      WRITE(6,6300)                                                     00009980
      II=I                                                              00009990
      DO 300 IZZ=1,II                                                   00010000
      DO 300 L=1,J                                                      00010010
      WRITE(6,6400) IZZ,L,(U(K,L,IZZ),K=1,6)                            00010020
  300 CONTINUE                                                          00010030
      ISTART=1                                                          00010040
  310 CONTINUE                                                          00010050
      IF(KOUNT.EQ.1) IP=1                                               00010060
      XPLOT(IP)=XN                                                      00010070
      IF(KOUNT.EQ.1) XPLOT(IP)=XA
      IP3=IP+J
      DO 400 K=1,5
      IT=ITSTOR(K)
      DO 400 L=1,J
      ULAST(IP,L,IT)=U(2,L,IT)
      ULAST(IP3,L,IT)=U(5,L,IT)
  400 CONTINUE
      IP=IP+1
      RETURN
 6300 FORMAT(//2X,*IZ*,3X,*L*, 7X,*F*,12X,*UF*,10X,*DOF*,11X,*G*,12X,
     X*DG*,10X,*DUG*//)
 6400 FORMAT(2I4,6(1X,E12.5))
      END
      SUBROUTINE ZMESH
C
C    THIS SUBROUTINE ALLOWS THE USER TO SUPPLY HIS OWN Z-MESH.  HE
C    SHOULD ALSO SUPPLY VALUES FOR THE FOLLOWING VARIABLES
C                    I---NUMBER OF Z-POINTS (MAXIMUM ALLOWED IS 11.
C                    OTHERWISE, THE DIMENSION IN THE MAIN PROGRAM OF THE
C                    VARIABLES HKZ, USTORE, AND ULAST MUST BE INCREASED).
C                    ZA---MINIMUM VALUE OF Z, USUALLY ZA=0.0
C                    ZB---MAXIMUM VALUE OF Z
C    THE MESH IS DEFINED BY HZ(L) WHERE Z(L+1)=Z(L)+HZ(L) WITH Z(1)=ZA
C    AND Z(I)=ZB.
C    IN THE EXAMPLE BELOW A UNIFORM Z-MESH IS PRESCRIBED.
C
      COMMON /MESHZ/ HZ(11)
      COMMON /PARM4/ I,J,IZ
      COMMON /PARM5/ FACX,HKS,XA,XB,YA,YB,ZA,ZB,NX
      ZA=0.0
      ZB=1.0
      Z=ZA
      I=1
      DO 10 L=1,10
      HZ(L)=.1
      Z=Z+HZ(L)
      I=I+1
      IF(ABS(Z-ZB).LE.1.0E-5) GO TO 20
      IF(Z.GE.ZB) GO TO 20
   10 CONTINUE
   20 RETURN
      END
      SUBROUTINE WALJET(XSTART)
C
C    THIS SUBROUTINE COMPUTES THE SOLUTION OF THE 3-D WALL JET PROBLEM
C    AND THE 2-D ATTACHMENT LINE EQUATION.
C
      LOGICAL SEPRAT
      LOGICAL REL,XSUPLY,YSUPLY,ZSUPLY,REFINE,USUPLY
      LOGICAL OPTPT
      COMMON /SEPRAT/ SEPRAT
      COMMON /OPTION/ XSUPLY,YSUPLY,ZSUPLY,REFINE,USUPLY
      COMMON /WRTE/ XPRT,OPTPT,NPRINT /KPRT/ KPRT(1)
      COMMON /ULAST/ ULAST(6,101,1)
      COMMON /USTORE/ USTORE(6,101,1)
      COMMON /MESHX/ HKX(1) /MESHY/ H(1) /UT/ UT(6,1) /UTX/ UTX(6,1)
      COMMON /G/ G(6,1) /MESHZ/ HKZ(1)
      COMMON /PARM3/ MAXITS,EPSERR,KTER,REL,CUTOFF,MTEST
      COMMON /PARM4/ I,J,IZ
      COMMON /PARM5/ FACX,HKS,XA,XB,YA,YB,ZA,ZB,NX
      COMMON /PARM6/ KK,XN,HX,X,ZN,HZ,Z
      DIMENSION SAVUT(6,101)
      DIMENSION GSAVE(2,101)
C
      KASE2=2
      KASE3=3
      N=6
      NP=4
      NQ=2
      KXNP=1
      NXSAVE=NX
      SEPRAT=.FALSE.
C
C
C SET UP MESHES AND INITIAL PROFILE
C
      XN=XA
      KT=KSTART+1
C
      DO 100 L=1,J
      GSAVE(1,L)=G(1,L)
      GSAVE(2,L)=G(5,L)
      DO 100 IZZ=1,1
      DO 100 K=1,N
  100 ULAST(K,L,IZZ)=USTORE(K,L,IZZ)
      DO 2000 KOUNT=KT,NX
      KK=KOUNT
      HX=HKX(KOUNT)
      IF((XN+HX).GT.XB) HX=XB-XN
      XN=XN+HX
      X=XN-HX/2.
      WRITE(6,6000) KOUNT,XN,HX
C
C SOLUTION AT PREVIOUS STREAMWISE STATION IS STORED IN VECTOR ** UTX **
C AS A FIRST GUESS TO FIND VISCOSITY ,VALUE AT PREVIOUS STREAMWISE
C STATION IS USED
C
      DO 1500 IZZ=1,I
      IZ=IZZ
      IZM1=IZ-1
      IF(IZ.NE.1) GO TO 1090
      ZN=ZA
      Z=ZA
      IF(KK.GT.6) GO TO 950
C
C SET UP AVERAGING TO ELIMINATE UNNATURAL OSCILATIONS CAUSED BY
C ARBITRARY INITIAL CONDITIONS
C
      DO 900 L=1,J
      G(1,L)=GSAVE(1,L)
      G(5,L)=GSAVE(2,L)
      DO 900 K=1,N
      UT(K,L)=USTORE(K,L,1)
      ULAST(K,L,1)=.5*(ULAST(K,L,1)+USTORE(K,L,1))
  900 UTX(K,L)=ULAST(K,L,1)
      GO TO 975
  950 CONTINUE
      DO 1000 L=1,J
      G(1,L)=GSAVE(1,L)
      G(5,L)=GSAVE(2,L)
      DO 1000 K=1,N
      UT(K,L)=USTORE(K,L,1)
 1000 UTX(K,L)=USTORE(K,L,1)
  975 CONTINUE
C
C SOLVE ATTACHMENT LINE EQUATION
C
      CALL NEWTON(KASE2,N,NP,NH)
      DO 980 L=1,J
      GSAVE(1,L)=G(2,L)
      GSAVE(2,L)=G(6,L)
  980 CONTINUE
      GO TO 1159
 1090 CONTINUE
      HZ=HKZ(IZM1)
      ZN=ZN+HZ
      Z=ZN-.5*HZ
      IF(KK.GT.6) GO TO 1095
      DO 1091 L=1,J
      G(1,L)=G(2,L)
      G(5,L)=G(6,L)
      DO 1091 K=1,N
      SAVUT(K,L)=UT(K,L)
      UTX(K,L)=.5*(UTX(K,L)+UT(K,L))
      IF(IZ.EQ.2) UTX(K,L)=UT(K,L)
      ULAST(K,L,IZ)=.5*(USTORE(K,L,IZ)+ULAST(K,L,IZ))
      USTORE(K,L,IZ)=ULAST(K,L,IZ)
 1091 CONTINUE
      GO TO 1150
 1095 CONTINUE
      DO 1140 I=1,J
      DO 1100 K=1,N
 1130 UTX(K,L)=UT(K,L)
      G(5,L)=G(6,L)
 1140 G(1,L)=G(7,L)
 1150 CONTINUE
C
C SOLVE 3-D WALL JET EQUATION
C
      CALL NEWTON(KASE3,N,NP,NU)
 1159 CONTINUE
C
C EXIT IF NO CONVERGENCE
C
      IF(KTER.EQ.0) GO TO 2100
      IF(IZ.EQ.1) GO TO 1495
      IF(KK.GT.6) GO TO 1485
      DO 1480 K=1,N
      DO 1480 L=1,J
 1480 USTORE(K,L,IZM1)=SAVUT(K,L)
      GO TO 1500
 1485 CONTINUE
      DO 1490 K=1,N
      DO 1490 L=1,J
 1490 USTORE(K,L,IZM1)=UTX(K,L)
      GO TO 1500
 1495 IF(KK.GT.6) GO TO 1500
      DO 1498 K=1,N
      DO 1498 L=1,J
 1498 USTORE(K,L,1)=UTX(K,L)
 1500 CONTINUE
      DO 1510 K=1,N
      DO 1510 L=1,J
 1510 USTORE(K,L,IZ)=UT(K,L)
C
C CHECK PRINT OUT OPTIONS
C
      IF(.NOT.OPTPT.AND.(XN.GE.KXNP*XPRI)) GO TO 1200
      IF(OPTPT.AND.(KOUNT.EQ.KPRT(KXNP))) GO TO 1200
      GO TO 1300
 1200 CONTINUE
      KXNP=KXNP+1
 1250 CONTINUE
C
C WRITE SOLUTION ON PAPER
C
      CALL OUTPT
 1300 CONTINUE
C
C
      IF(UT(3,1).GE.0.) GO TO 2050
C
C SEPARATION
```

```
C
      WRITE(6,6200)
      SEPRAT=.TRUE.
      NXSAVE=KOUNT-1
      GO TO 2100
 2050 CONTINUE
      IF(XN.GE.XH) GO TO 2100
 2000 CONTINUE
 2100 CONTINUE
C
 6000 FORMAT(* STATION = *,I4,*, X =*,E12.5,*, X-STEP =*,E12.5)
 6100 FORMAT(* O(H2) SOLUTION*)
 6200 FORMAT(///*XXXXXXXXXXXXXX SEPARATION XXXXXXXXXXXXXX*///)
C
      CALL OUTPT
      NX=NXSAVE
      IF(KTER.NE.0) GO TO 2200
      SEPRAT=.TRUE.
      NX=KK-1
 2200 RETURN
      END
      SUBROUTINE PRMESH
C
C    THIS ROUTINE ALLOWS THE USER TO SUPPLY THE STREAMWISE STATIONS AT
C    WHICH SOLUTIONS ARE TO BE PRINTED ON PAPER.  THE FIRST STATION
C    (INITIAL CONDITIONS) IS AUTOMATICALLY PRINTED BY THE PROGRAM.
C    THEREFORE, KPRT(L) .GT. 1 AND SHOULD HAVE THE PROPERTY THAT KPRT(L)
C    IS GREATER THAN KPRT(K) FOR L GREATER THAN K.  KPRT(M)=I MEANS THAT
C    THE MTH PRINT OUT WILL BE FOR THE PLANE X=X(I).  NOTE ALSO THAT THE
C    LAST STATION IS AUTOMATICALLY PRINTED BY THE PROGRAM, SO IT NEED NOT
C    BE INCLUDED IN THE VECTOR KPRT.
C    IN THE SAMPLE CASE BELOW, PRINTING OF THE SOLUTION OCCURS AT THE
C    FIRST, LAST, AND FORTIETH STREAMWISE STATION.
C
      COMMON /KPRT/ KPRT(1)
      KPRT(1)=40
      KPRT(2)=-1
      RETURN
      END
      SUBROUTINE YMESH
C
C    THIS SUBROUTINE ALLOWS THE USER TO SUPPLY HIS OWN ETA MESH.  HE
C    SHOULD ALSO SUPPLY VALUES FOR THE FOLLOWING VARIABLES
C                 J---TOTAL NUMBER OF ETA POINTS (MAXIMUM OF 101)
C                 YA---MINIMUM VALUE OF ETA (USUALLY YA=0.0)
C                 YB---MAXIMUM VALUE OF ETA
C    THE MESH IS DEFINED BY HY(L) WHERE ETA(L+1)=ETA(L)+HY(L) WITH
C    ETA(1)=YA AND ETA(J)=YB.
C    THE SAMPLE EXAMPLE STARTS WITH A GEOMETRIC STRETCHING UNTIL HY(L) IS
C    .GT..5.  IT THEN REMAINS FIXED AT .5 UNTIL L.GT.50 WHEN IT IS SET TO
C    .7.
```

```
C
      COMMON /MESHY/ HY(1)
      COMMON /PARM4/ I,J,IZ
      COMMON /PARMS/ FACX,HKS,XA,XB,YA,YB,ZA,ZB,NX
      YA=0.0
      YB=60.0
      J=2
      HY(1)=.005
      Y=HY(1)
      DO 10 L=2,100
      HY(L)=1.2*HY(L-1)
      IF(HY(L).GT..50) HY(L)=.5
      IF(L.GT.50) HY(L)=.75
      Y=Y+HY(L)
      J=J+1
      IF(ABS(Y-YB).LE.1.0E-5) GO TO 20
      IF(Y.GE.YB) GO TO 20
   10 CONTINUE
   20 YB=Y
      RETURN
      END
      SUBROUTINE XMESH
C
C    THIS SUBROUTINE ALLOWS THE USER TO SUPPLY HIS OWN X-MESH.  HE
C    SHOULD ALSO SUPPLY VALUES FOR THE FOLLOWING VARIABLES
C                   NX---TOTAL NUMBER OF X-POINTS MINUS ONE (MAX 401)
C                   XA---INITIAL X-PLANE
C                   XB---FINAL X-PLANE (MAXIMUM VALUE OF X)
C    THE MESH IS DEFINED BY HX(L) WHERE X(L)=X(L-1)+HX(L) WITH X(1)=XA
C    AND X(NX)=XB.  NOTE ONLY HX(2),HX(3),....,HX(NX) NEED BE LOADED.
C    HX(NX) IS ALWAYS AUTOMATICALLY DECREASED SO THAT X(NX)=XB.
C    IN THE EXAMPLE BELOW HX IS SET TO .0001 FOR THE FIRST 9 STEPS THAN A
C    GEOMETRIC STRETCHING ALGORITHM INCREASES THE NET UNTIL XB IS REACHED
C
      COMMON /MESHX/ HX(1)
      COMMON /PARMS/ FACX,HKS,XA,XB,YA,YB,ZA,ZB,NX
      XA=1.0
      XB=10.0
      HX(1)=.0001
      X=XA
      DO 10 L=2,401
      NX=L
      HX(L)=1.2*HX(L-1)
      IF(L.LT.11) HX(L)=.0001
      X=X+HX(L)
      IF(ABS(X-XB).LE.1.0E-5) GO TO 20
      IF(X.GE.XB) GO TO 20
   10 CONTINUE
   20 RETURN
      END
      SUBROUTINE [illegible]
```

```
C                                                                       
C NET SELECTION - APPROXIMATELY CHOOSING H(K) WITH THE TRUNCATION ERROR
C TO BE A CONSTANT ON THE WHOLE INTERVAL
C
      COMMON /DU/ US(6,1) /UT/ UT(6,1) /HNEW/ HNEW(1) /MESHY/ H(1)
      COMMON /F/ F(6,1)
      COMMON /NET/ JMAX,HMAX,IFMX,KSAME
      COMMON /PARM1/ N,NP,NO
      COMMON /PARM4/ I,J,IZ
      COMMON /PARM5/ FACX,HKS,XA,XB,YA,YB,ZA,ZB,NX
      COMMON /NET1/ KTYPE,KSINGK,KSING(1),KL,P(6),Q(6),R(6),TAU2(6)
      COMMON /SETUP/ UH(6),SUM(6),UD1(6),UD2(6),FF(6),AJA(6,6)
      DIMENSION Z(2000),KT(10)
      LOGICAL DEL1,DEL2
C
      KSINGK=1
      KSING(1)=1
      KEXCED=0
      NETINC=0
      AREA=0.
      KSAME=0
      IFMX=1
      J1=J-1
      DO 10 K=1,N
      SUM(K)=0.
   10 CONTINUE
C
C COMPUTE LOCAL TRUNCATION ERROR AT MID-POINT
C
      KTYPE=-1
      DO 300 L=1,J1
      KL=L
      IF(L.EQ.1) GO TO 100
      KTYPE=0
      IF(L.EQ.KSING(KSINGK)) GO TO 50
      IF(L.NE.(KSING(KSINGK)-1)) GO TO 100
C
C POINT BEFORE SINGULARITY OR RIGHT END-POINT
C
      KTYPE=1
      GO TO 100
   50 CONTINUE
C
C POINT AFTER SINGULARITY
C
      KTYPE=-1
      KSINGK=KSINGK+1
  100 CONTINUE
      DO 150 K=1,N
      UH(K)=(UT(K,L)+UT(K,L+1))/2.
  150 CONTINUE
```

```
      CALL TRUN
      T=0.
      DO 200 K=1,N
      F(K,L)=TAU2(K)
      T=T+TAU2(K)**2
  200 CONTINUE
      T=SQRT(T)
      Z(L)=T
      AREA=AREA+H(L)**2*T/2.
  300 CONTINUE
C
      CONSTC=AREA/J1
C
C NET SELECTION
C
      LA=1
      DO 2000 KOUT=1,KSINGK
      LB=KSING(KOUT)-1
      IF(KOUT.GT.1) LA=KSING(KOUT)
      DEL1=.FALSE.
      DEL2=.FALSE.
      DO 1000 L=LA,LB
      FACT=H(L)**2*Z(L)/(2.*CONSTC)
      FACT=SQRT(FACT)
      Z(L)=FACT
      KJS=FACT+0.5
      IF(KJS.LE.1) GO TO 700
C
C ADDITION
C
      IF((L+KJS+NETINC+1).GT.JMAX) GO TO 2500
      KSAME=1
      IFMX=MAX0(IFMX,KJS)
      HI=H(L)/KJS
      DO 600 M=1,KJS
      HNEW(NETINC+M-1+L)=HI
      DO 600 K=1,N
      US(K,NETINC+M-1+L)=((M-1)*UT(K,L+1)+(KJS-(M-1))*UT(K,L))/KJS
  600 CONTINUE
      NETINC=NETINC+KJS-1
      DEL1=.FALSE.
      DEL2=.FALSE.
      GO TO 1000
  700 CONTINUE
C
      IF(FACT.GE.0.5) GO TO 800
      DEL2=.TRUE.
      IF(DEL1) GO TO 900
  800 CONTINUE
C
C NO DELETION
```

```
C                                                                        00014150
      IF((L+NETINC+1).GT.JMAX) GO TO 2500                                00014160
      DEL1=DEL2                                                          00014170
      DO 850 K=1,N                                                       00014180
      US(K,L+NETINC)=UT(K,L)                                             00014190
  850 CONTINUE                                                           00014200
      HNEW(L+NETINC)=H(L)                                                00014210
      GO TO 1000                                                         00014220
  900 CONTINUE                                                           00014230
C                                                                        00014240
C DELETION                                                               00014250
C                                                                        00014260
      DEL1=DEL2                                                          00014270
      IF((HNEW(L+NETINC-1)+H(L)).LE.HMAX) GO TO 825                      00014280
      KEXCED=1                                                           00014290
      GO TO 800                                                          00014300
  825 CONTINUE                                                           00014310
      KSAME=1                                                            00014320
      HNEW(L+NETINC-1)=HNEW(NETINC+L-1)+H(L)                             00014330
      NETINC=NETINC-1                                                    00014340
 1000 CONTINUE                                                           00014350
      KT(KOUT)=KSING(KOUT)+NETINC                                        00014360
 2000 CONTINUE                                                           00014370
      DO 2100 K=1,KSINGK                                                 00014380
      KSING(K)=KT(K)                                                     00014390
 2100 CONTINUE                                                           00014400
      IF(KSAME.EQ.0) RETURN                                              00014410
C                                                                        00014420
      JNEW=J+NETINC                                                      00014430
      DO 2200 K=1,N                                                      00014440
      US(K,JNEW)=UT(K,J)                                                 00014450
 2200 CONTINUE                                                           00014460
      J=JNEW                                                             00014470
      HNEW(J)=0.                                                         00014480
      DO 2300 L=1,J                                                      00014490
      H(L)=HNEW(L)                                                       00014500
      DO  300 K=1,N                                                      00014510
      UT(K,L)=US(K,L)                                                    00014520
 2300 CONTINUE                                                           00014530
      RETURN                                                             00014540
C                                                                        00014550
 2500 CONTINUE                                                           00014560
      KSAME=0                                                            00014570
C                                                                        00014580
      RETURN                                                             00014590
      END                                                                00014600
      SUBROUTINE TRUN                                                    00014610
C                                                                        00014620
C THIS SUBROUTINE COMPUTES THE LOCAL TRUNCATION ERROR OF THE BOX          00014630
C SCHEME. THE INTEGER VARIABLE ** TYPE ** INDICATES THE FOLLOWING         00014640
C          TYPE =  -1  LEFT BOUNDARY POINT                                00014650
```

```
C               =    0   INTERNAL POINT
C               =    1   RIGHT BOUNDARY POINT
C SINGULAR POINTS ARE TREATED AS BOUNDARY POINTS
C
      INTEGER TYPE
      COMMON /MESHY/ H(1) /UT/ UT(6,1)
      COMMON /PARMI/ N,NP,NO
      COMMON /PARMS/ FACX,HKS,XA,XB,YA,YB,ZA,ZB,NX
      COMMON /NETI/ TYPE,KSINGK,KSING(1),L,P(6),Q(6),R(6),T(6)
      COMMON /SETUP/ UH(6),UHX(6),UD1(6),UD2(6),FF(6),A(6,6)
C
      IF(TYPE) 100, 200, 300
  100 CONTINUE
C
C LEFT BOUNDARY POINT
C
      A1=-H(L)/2.
      A2=-A1
      A3=A2+H(L+1)
      A4=A3+H(L+2)
      A5=A4+H(L+3)
C
      L1=L
      L2=L+1
      L3=L+2
      L4=L+3
      L5=L+4
C
      GO TO 400
  200 CONTINUE
C
C INTERNAL POINT
C
      IF(L.EQ.(KSING(KSINGK)-2)) GO TO 250
C
      A1=-H(L-1)-H(L)/2.
      A2=-H(L)/2.
      A3=H(L)/2.
      A4=A3+H(L+1)
      A5=A4+H(L+2)
C
      L1=L-1
      L2=L
      L3=L+1
      L4=L+2
      L5=L+3
C
      GO TO 400
  250 CONTINUE
C
C SPECIAL TREATMENT FOR L = (KSING(KSINGK) - 2) BECAUSE OF THIRD DERIVAT
```

```
C                                                                       00015170
      A1=H(L)/2.+H(L+1)                                                 00015180
      A2=H(L)/2.                                                        00015190
      A3=-A2                                                            00015200
      A4=A3-H(L-1)                                                      00015210
      A5=A4-H(L-2)                                                      00015220
C                                                                       00015230
      L1=L+1                                                            00015240
      L2=L                                                              00015250
      L3=L-1                                                            00015260
      L4=L-2                                                            00015270
      L5=L-3                                                            00015280
C                                                                       00015290
      GO TO 400                                                         00015300
  300 CONTINUE                                                          00015310
C                                                                       00015320
C RIGHT BOUNDARY POINT                                                  00015330
C                                                                       00015340
      A1=H(L)/2.                                                        00015350
      A2=-A1                                                            00015360
      A3=A2-H(L-1)                                                      00015370
      A4=A3-H(L-2)                                                      00015380
      A5=A4-H(L-3)                                                      00015390
C                                                                       00015400
      L1=L+1                                                            00015410
      L2=L                                                              00015420
      L3=L-1                                                            00015430
      L4=L-2                                                            00015440
      L5=L-3                                                            00015450
C                                                                       00015460
  400 CONTINUE                                                          00015470
C                                                                       00015480
C COMPUTE LOCAL TRUNCATION ERROR IN TWO STEPS                           00015490
C (A) CONTRIBUTION FROM THIRD DERIVATIVE                                00015500
C                                                                       00015510
      UP=VANDET(5,A1,A2,A3,A4,A5)                                       00015520
      C1=6.*(VANDET(3,A3,A4,A5,0.,0.)*A2**4                             00015530
     1    -VANDET(3,A2,A4,A5,0.,0.)*A3**4                               00015532
     2    +VANDET(3,A2,A3,A5,0.,0.)*A4**4                               00015534
     3    -VANDET(3,A2,A3,A4,0.,0.)*A5**4)/UP                           00015536
      C2=-6.*(VANDET(3,A3,A4,A5,0.,0.)*A1**4                            00015540
     1    -VANDET(3,A1,A4,A5,0.,0.)*A3**4                               00015542
     2    +VANDET(3,A1,A3,A5,0.,0.)*A4**4                               00015544
     3    -VANDET(3,A1,A3,A4,0.,0.)*A5**4)/UP                           00015546
      C3=6.*(VANDET(3,A2,A4,A5,0.,0.)*A1**4                             00015550
     1    -VANDET(3,A1,A4,A5,0.,0.)*A2**4                               00015552
     2    +VANDET(3,A1,A2,A5,0.,0.)*A4**4                               00015554
     3    -VANDET(3,A1,A2,A4,0.,0.)*A5**4)/UP                           00015556
      C4=-6.*(VANDET(3,A2,A3,A5,0.,0.)*A1**4                            00015560
     1    -VANDET(3,A1,A3,A5,0.,0.)*A2**4                               00015562
     2    +VANDET(3,A1,A2,A5,0.,0.)*A3**4                               00015564
     3    -VANDET(3,A1,A2,A3,0.,0.)*A5**4)/UP                           00015566
      C5=6.*(VANDET(3,A2,A3,A4,0.,0.)*A1**4                             00015570
     1    -VANDET(3,A1,A3,A4,0.,0.)*A2**4                               00015572
     2    +VANDET(3,A1,A2,A4,0.,0.)*A3**4                               00015574
     3    -VANDET(3,A1,A2,A3,0.,0.)*A4**4)/UP                           00015576
      DO 500 K=1,N                                                      00015580
      P(K)=C1*UT(K,L1)+C2*UT(K,L2)+C3*UT(K,L3)+C4*UT(K,L4)+C5*UT(K,L5)  00015590
  500 CONTINUE                                                          00015600
C                                                                       00015610
C (d) CONTRIBUTION FROM SECOND DERIVATIVE                               00015620
C                                                                       00015630
      UP=VANDET(4,A1,A2,A3,A4,0.)                                       00015640
      C1=2.*(VANDET(2,A3,A4,0.,0.,0.)*A2**3                             00015650
     1    -VANDET(2,A2,A4,0.,0.,0.)*A3**3                               00015652
     2    +VANDET(2,A2,A3,0.,0.,0.)*A4**3)/UP                           00015654
      C2=-2.*(VANDET(2,A3,A4,0.,0.,0.)*A1**3                            00015660
     1    -VANDET(2,A1,A4,0.,0.,0.)*A3**3                               00015662
     2    +VANDET(2,A1,A3,0.,0.,0.)*A4**3)/UP                           00015664
      C3=2.*(VANDET(2,A2,A4,0.,0.,0.)*A1**3                             00015670
     1    -VANDET(2,A1,A4,0.,0.,0.)*A2**3                               00015672
     2    +VANDET(2,A1,A2,0.,0.,0.)*A4**3)/UP                           00015674
      C4=-2.*(VANDET(2,A2,A3,0.,0.,0.)*A1**3                            00015680
     1    -VANDET(2,A1,A3,0.,0.,0.)*A2**3                               00015682
     2    +VANDET(2,A1,A2,0.,0.,0.)*A3**3)/UP                           00015684
C                                                                       00015690
      DO 600 K=1,N                                                      00015700
      R(K)=C1*UT(K,L1)+C2*UT(K,L2)+C3*UT(K,L3)+C4*UT(K,L4)              00015710
  600 CONTINUE                                                          00015720
      CALL RHSF2D                                                       00015730
C                                                                       00015740
      DO 800 K=1,N                                                      00015750
      U(K)=0.                                                           00015760
      DO 700 M=1,N                                                      00015770
      U(K)=U(K)+A(K,M)*R(M)                                             00015780
  700 CONTINUE                                                          00015790
  800 CONTINUE                                                          00015800
C                                                                       00015810
C SECOND ORDER TRUNCATION ERROR                                         00015820
C                                                                       00015830
      DO 900 K=1,N                                                      00015840
      F(K)=P(K)-3.*U(K)                                                 00015850
  900 CONTINUE                                                          00015860
C                                                                       00015870
      RETURN                                                            00015880
      END                                                               00015890
      FUNCTION VANDET(N,X1,X2,X3,X4,X5)                                 00015900
C                                                                       00015910
C THIS FUNCTION ROUTINE COMPUTES THE DETERMINANT OF AN (N * N)          00015920
C VANDERMONDE MATRIX WITH ENTRIES X(1),X(2),......, WITH N GREATER THAN 00015930
C 3 AND LESS THAN 7                                                     00015940
C                                                                       00015950
```

```
      DIMENSION X(5)
C
      X(1)=X1
      X(2)=X2
      X(3)=X3
      X(4)=X4
      X(5)=X5
      N1=N-1
      VANDET=1.
      DO 200 L=1,N1
      L1=L+1
      DO 100 K=L1,N
      VANDET=VANDET*(X(K)-X(L))
  100 CONTINUE
  200 CONTINUE
C
C
      RETURN
      END
      SUBROUTINE PROFLE
C
C    THIS SUBROUTINE SETS UP THE INITIAL PROFILES AND CALLS THE MESH
C    REFINEMENT ROUTINE IF REFINE=.TRUE..  THE USER MAY CHANGE THE
C    PROFILES BY LOADING NEW VALUES INTO THE MATRIX USTORE.
C
      LOGICAL REFINE
      COMMON /OPTION/ XSUPLY,YSUPLY,ZSUPLY,REFINE,USUPLY
      COMMON /UTX/ UTX(6,1)
      COMMON /UT/ UT(6,1)
      COMMON /MESHY/ H(1)
      COMMON /MESHX/ HKX(1)
      COMMON /NET/ JMAX,NMAX,IFMX,KSAME
      COMMON /PARM6/ I,J,IZ
      COMMON /PARM5/ FACX,HKS,XA,XB,YA,YB,ZA,ZB,NX
      COMMON /PARM4/ KOUNT,XN,HX,X,ZN,HZ,Z
      COMMON /USTORE/ USTORE(6,101,1)
      COMMON /ETA/ ETA(1)
      DATA ISTART/0/
      Y1=1.0
      E1=EXP(-Y1)
      H(J)=0.0
    1 CONTINUE
      DO 30 M=1,I
      Y=YA
      SUM=0.0
C
C    LOAD INITIAL PROFILES
C
      USTORE(1,1,M)=0.0
      A=1.5+.1*FLOAT(M-1)
      IF(M.GT.6) A=2.0-.1*FLOAT(M-6)
      C2=A/(E1+Y1*E1-1.0)                                             00016470
      C3=C2*E1                                                        00016480
      DO 30 L=1,J                                                     00016490
      ETA(L)=Y                                                        00016500
      IF(Y.GT.Y1) GO TO 10                                            00016510
      E=EXP(-Y)                                                       00016520
      USTORE(2,L,M)=C3*Y+C2*(E-1.0)                                   00016530
      USTORE(3,L,M)=C3-C2*E                                           00016540
      GO TO 15                                                        00016550
   10 E=EXP(-.1*(Y-Y1)**2)                                            00016560
      USTORE(2,L,M)=1.0+(A-1.0)*E                                     00016570
      USTORE(3,L,M)=-.2*(Y-Y1)*(A-1.0)*E                              00016580
   15 T=USTORE(2,L,M)                                                 00016590
      IF(L.EQ.1) GO TO 18                                             00016600
      SUM=SUM+.5*H(L-1)*(T+T1)                                        00016610
      USTORE(1,L,M)=SUM                                               00016620
   18 T1=T                                                            00016630
   20 USTORE(4,L,M)=USTORE(1,L,M)                                     00016640
      USTORE(5,L,M)=USTORE(2,L,M)                                     00016650
      USTORE(6,L,M)=USTORE(3,L,M)                                     00016660
      Y=Y+H(L)                                                        00016670
   30 CONTINUE                                                        00016680
      KOUNT=1                                                         00016690
      IF(ISTART.EQ.0) GO TO 999                                       00016700
      WRITE(6,6000)                                                   00016710
      CALL OUTPT                                                      00016720
      RETURN                                                          00016730
  999 CONTINUE                                                        00016740
      DO 60 L=1,J                                                     00016750
      DO 60 K=1,6                                                     00016760
      UTX(K,L)=USTORE(K,L,1)                                          00016770
   60 UT(K,L)=USTORE(K,L,1)                                           00016780
      IF(.NOT.REFINE) RETURN                                          00016790
C                                                                     00016800
C   SET UP MESH REFINEMENT FOR THE ABOVE PROFILES                     00016810
C                                                                     00016820
      KOUNT=2                                                         00016830
      HX=HKX(KOUNT)                                                   00016840
      XN=XA+HX                                                        00016850
      X=XA+.5*HX                                                      00016860
      Z=0.0                                                           00016870
      ZN= .0                                                          00016880
      HZ=0.0                                                          00016890
      IZ=1                                                            00016900
      CALL NEWTON(2,6,4,2)                                            00016910
      CALL METHOD                                                     00016920
   70 ISTART=1                                                        00016930
      GO TO 1                                                         00016940
 6000 FORMAT(1H0,10X,*INITIAL PROFILES*)                              00016950
 6100 FORMAT(* PROFLF--NO CHANGE IN REFINEMENT*)                      00016960
      END                                                             00016970
      SUBROUTINE PREPG2
C
C     PREVIOUS STATION FOR ATTACHMENT LINE EQUATION
C
      COMMON /UTX/ UX(6,1) /G/ G(6,1) /MESHY/ HY(1)
      COMMON /PARM1/ N,NP,NO
      COMMON /PARM2/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10
      COMMON /PARM4/ I,J,IZ
      COMMON /PARM6/ KOUNT,XN,HX,X,ZN,HZ,Z
      DIMENSION UH(6)
C
C     COMPUTE P COEFFICIENTS
C
      CALL PREPP(XN-HX,Z,8)
      ALPHAN=P10/HX
      DO 100 L=2,J
      L1=L-1
      H=HY(L1)
      DO 10 K=1,N
   10 UH(K)=.5*(UX(K,L)+UX(K,L1))
      G(3,L)=G(1,L)*UX(3,L)-G(1,L1)*UX(3,L1)-H*((-ALPHAN+P2)*UH(2)*UH(2)
     X         +(ALPHAN-P1)*UH(1)*UH(3)-P3*UH(4)*UH(3)-P2)
      G(4,L)=G(5,L)*UX(6,L)-G(5,L1)*UX(6,L1)+H*((ALPHAN-P4)*UH(2)*UH(5)
     X         +(-ALPHAN+P1)*UH(1)*UH(6)+P3*(1.0-UH(5)*UH(5)+UH(4)*UH(6))
     X         +P4)
  100 CONTINUE
      RETURN
      END
      SUBROUTINE PREPG
C
C     PREVIOUS STATION FOR FULL 3-D WALL JET EQUATION
C
      COMMON /MESHY/ HY(1) /UTX/ UX(6,1) /G/ G(6,1)
      COMMON /USTORE/ US(6,101,1)
      COMMON /PARM1/ N,NP,NO
      COMMON /PARM2/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10
      COMMON /PARM4/ I,J,IZ
      COMMON /PARM6/ KOUNT,XN,HX,X,ZN,HZ,Z
      DIMENSION UHX(6),UHXX(6),UHXXX(6),UI(6)
      IZM1=IZ-1
C
C     COMPUTE P COEFFICIENTS
C
      CALL PREPP(X,Z,1)
      TERM=2.0*P10
      PROD1=TERM/HX
      PROD2=TERM*P7/HZ
      DO 100 L=2,J
      L1=L-1
      H=HY(L1)
      DO 10 K=1,N
      UHX(K)=.5*(UX(K,L)+UX(K,L1))                                      00017460
      UHXX(K)=.5*(US(K,L,IZ)+US(K,L1,IZ))                               00017470
      UHXXX(K)=.5*(US(K,L,IZM))+US(K,L1,IZM))                           00017480
      UT(K)=UHX(K)+UHXX(K)+UHXXX(K)                                     00017490
   10 CONTINUE                                                          00017500
      G(3,L)=G(1,L)*(UX(3,L)+US(3,L,IZ)+US(3,L,IZM))                    00017510
     X       -G(1,L1)*(UX(3,L1)+US(3,L1,IZM))+US(3,L1,IZ))+             00017512
     X       .25*H*(P1*UT(1)*UT(3)-P2*UT(2)*UT(2)-P5*UT(2)*UT(5)+       00017514
     X       P6*UT(3)*UT(4)-P8*UT(5)*UT(5)-PROD1*(UHX(2)*UHX(2)-(UHXX(2)00017516
     X       +UHXXX(2))**2-UT(3)*(UHX(1)-UHXX(1)-UHXXX(1)))-PROD2*(UT(5)00017518
     X       *(UHXX(2)-UHX(2)-UHXXX(2))-UT(3)*(UHXX(4)-UHX(4)-UHXXX(4)))00017520
     X       )                                                          00017522
      G(4,L)=G(5,L)*(UX(6,L)+US(6,L,IZ)+US(6,L,IZM)))                   00017530
     X       -G(5,L1)*(UX(6,L1))+US(6,L1,IZ)+US(6,L1,IZM)))             00017532
     X       +.25*H*(P1*UT(1)*UT(6)-P4*UT(2)*UT(5)-P3*UT(5)*UT(5)       00017534
     X       +P6*UT(4)*UT(6)-P9*UT(2)*UT(2)-PROD1*(UT(2)*(UHX(5)-UHXX(5)00017536
     X       -UHXXX(5))-UT(6)*(UHX(1)-UHXX(1)-UHXXX(1)))-PROD2*(UHXX(5)*00017538
     X       *2-(UHX(5)+UHXXX(5))**2-UT(6)*(UHXX(4)-UHX(4)-UHXXX(4))))  00017540
  100 CONTINUE                                                          00017550
      RETURN                                                            00017560
      END                                                               00017570
      SUBROUTINE RHSF2D                                                 00017580
C                                                                       00017590
C     RIGHT-HAND SIDE FUNCTION FOR ATTACHMENT LINE EQUATION             00017600
C                                                                       00017610
      COMMON /SETUP/ U(6),UX(6),UD1(6),UDZ(6),F(6),A(6,6)               00017620
      COMMON /PARM2/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10                     00017630
      COMMON /PARM6/ KOUNT,XN,HX,X,ZN,HZ,Z                              00017640
      F(1)=U(2)                                                         00017650
      F(2)=U(3)                                                         00017660
      F(4)=U(5)                                                         00017670
      F(5)=U(6)                                                         00017680
      ALPHAN=P10/HX                                                     00017690
      F(3)=-(P1+ALPHAN)*U(1)*U(3)+(P2+ALPHAN)*U(2)*U(2)-P3*U(4)*U(3)    00017700
     X      -ALPHAN*(U(1)*UX(3)-U(3)*UX(1))-P2                          00017702
      F(6)=-P1*U(1)*U(6)+(P4+ALPHAN)*U(2)*U(5)+P3*(U(5)*U(5)-U(4)*U(6))+00017710
     X      ALPHAN*(UX(2)*U(5)-UX(5)*U(2)-U(1)*U(6)-UX(6)*U(1)+UX(1)*U(6)00017712
     X      )-P3-P4                                                     00017714
C                                                                       00017720
C     JACOBIAN MATRIX FOR ATTACHMENT LINE EQUATION                      00017730
C                                                                       00017740
      P1ALP=P1+ALPHAN                                                   00017750
      A(1,2)=1.0                                                        00017760
      A(2,3)=1.0                                                        00017770
      A(4,5)=1.0                                                        00017780
      A(5,6)=1.0                                                        00017790
      A(3,1)=-P1ALP*U(3)-ALPHAN*UX(3)                                   00017800
      A(3,2)=2.0*(P2+ALPHAN)*U(2)                                       00017810
      A(3,3)=-P1ALP*U(1)-P3*U(4)+ALPHAN*UX(1)                           00017820
      A(3,4)=-P3*U(3)                                                   00017830
      A(6,1)=-P1ALP*U(6)-ALPHAN*UX(6)                                   00017840
      A(6,2)=(P4+ALPHAN)*U(5)-ALPHAN*UX(5)                              00017850
      A(6,4)=-P3*U(6)                                                   00017860
      A(6,5)=(P4+ALPHAN)*U(2)+2.0*P3*U(5)+ALPHAN*UX(2)                  00017870
      A(6,6)=-P1ALP*U(1)-P3*U(4)+ALPHAN*UX(1)                           00017880
      RETURN                                                            00017890
      END                                                               00017900
      SUBROUTINE RHSF                                                   00017910
C                                                                       00017920
C     RIGHT-HAND SIDE FUNCTION                                          00017930
C                                                                       00017940
      COMMON /SETUP/ U(6),UX(6),UXX(6),UXXX(6),F(6),A(6,6)              00017950
      COMMON /PARM1/ N,NP,NQ                                            00017960
      COMMON /PARM2/ P1,P2,P3,P4,P5,P6,P7,P8,P9,P10                     00017970
      COMMON /PARM6/ KOUNT,XN,HX,X,ZN,HZ,Z                              00017980
      DIMENSION UT(6),UBAR(6)                                           00017990
      DO 10 K=1,N                                                       00018000
      UT(K)=.25*(UX(K)+UXX(K)+UXXX(K))                                  00018010
      UBAR(K)=.75*U(K)+UT(K)                                            00018020
   10 CONTINUE                                                          00018030
      F(1)=U(2)                                                         00018040
      F(2)=U(3)                                                         00018050
      F(4)=U(5)                                                         00018060
      F(5)=U(6)                                                         00018070
      TERM=.5*P10                                                       00018080
      PROD1=TERM/HX                                                     00018090
      PROD2=TERM*P7/HZ                                                  00018100
      F(3)=-P1*(U(1)*UBAR(3)+U(3)*UT(1))+P2*U(2)*(.25*U(2)+2.0*UT(2))   00018110
     X      +P5*(U(2)*UBAR(5)+U(5)*UT(2))-P6*(U(3)*UBAR(4)+U(4)*UT(3))  00018112
     X      +P8*U(5)*(.25*U(5)+2.0*UT(5))-4. *(P2+P5+P8)                00018114
     X      +PROD1*(U(2)*(U(2)+2.0*UX(2))-U(3)*(U(1)+UX(1)-UXX(1)-UXXX(100018116
     X      )-4.0*U(1)*UT(3))                                           00018118
     X      +PROD2*(U(5)*(U(2)+UXX(2)-UX(2)-UXXX(2))+4.0*U(2)*UT(5)     00018120
     X      -U(3)*(U(4)+UXX(4)-UX(4)-UXXX(4))-4.0*U(4)*UT(3))           00018122
      F(6)=-P1*(U(1)*UBAR(6)+U(6)*UT(1))+P4*(U(2)*UBAR(5)+U(5)*UT(2))   00018130
     X      +P3*U(5)*(.25*U(5)+2.0*UT(5))-P6*(U(4)*UBAR(6)+U(6)*UT(4))  00018132
     X      +P9*U(2)*(.25*U(2)+2.0*UT(2))-4. *(P4+P3+P9)                00018134
     X      +PROD1*(U(2)*(U(5)+UX(5)-UXX(5)-UXXX(5))+4.0*(U(5)*UT(2)-U(100018136
     X      *UT(6))-U(6)*(U(1)+UX(1)-UXX(1)-UXXX(1)))                   00018138
     X      +PROD2*(U(5)*(U(5)+2.0*UXX(5))-U(6)*(U(4)+UXX(4)-UX(4)      00018140
     X      -UXXX(4))-4.0*U(4)*UT(6))                                   00018142
C                                                                       00018150
C     JACOBIAN OF RIGHT-HAND SIDE FUNCTION                              00018160
C                                                                       00018170
      PROD1=4.0*PROD1                                                   00018180
      PROD2=4.0*PROD2                                                   00018190
      A(1,2)=1.0                                                        00018200
      A(2,3)=1.0                                                        00018210
      A(4,5)=1.0                                                        00018220
      A(5,6)=1.0                                                        00018230
      A(3,1)=-UBAR(3)*(P1+PROD1)                                        00018240
      A(3,2)=2.0*P2*UBAR(2)+UBAR(5)*(P5+PROD2)+.5*PROD1*(U(2)+UX(2))    00018250
      A(3,1)=-P1*UBAR(1)-P6*UBAR(4)-PROD1*(UBAR(1)-.5*(UXX(1)+UXXX(1)))  00018260
     X       -PROD2*(UBAR(4)-.5*(UX(4)+UXXX(4)))                          00018262
      A(3,4)=-(P6+PROD2)*UBAR(3)                                          00018270
      A(3,5)=P5*UBAR(2)+2.0*P8*UBAR(5)+PROD2*(UBAR(2)-.5*(UX(2)+UXXX(2))00018280
     X       )                                                             00018282
      A(6,1)=-(P1+PROD1)*UBAR(6)                                          00018290
      A(6,2)=P4*UBAR(5)+2.0*P9*UBAR(2)+PROD1*(UBAR(5)-.5*(UXX(5)+UXXX(5)00018300
     X       ))                                                            00018302
      A(6,4)=-(P6+PROD2)*UBAR(6)                                          00018310
      A(6,5)=(P4+PROD1)*UBAR(2)+2.0*P3*UBAR(5)+.5*PROD2*(U(5)+UXX(5))    00018320
      A(6,6)=-(P1+PROD1)*UBAR(1)-(P6+PROD2)*UBAR(4)                       00018330
     X       +.5*(PROD2*(UX(4)+UXXX(4))+PROD1*(UXX(1)+UXXX(1)))          00018332
      RETURN                                                              00018340
      END                                                                 00018350
      SUBROUTINE BC                                                       00018360
C                                                                         00018370
C    BOUNDARY CONDITION ROUTINE                                           00018380
C                                                                         00018390
      COMMON /SETUP/ UA(6),UB(6),UD1(6),UD2(6),G(6),B(6,6)               00018400
      G(1)=UA(1)                                                          00018410
      G(2)=UA(2)                                                          00018420
      G(3)=UA(4)                                                          00018430
      G(4)=UA(5)                                                          00018440
      G(5)=UB(2)-1.0                                                      00018450
      G(6)=UB(5)-1.0                                                      00018460
C                                                                         00018470
C    JACOBIAN OF THE BOUNDARY CONDITIONS                                  00018480
C                                                                         00018490
      B(1,1)=1.0                                                          00018500
      B(2,2)=1.0                                                          00018510
      B(3,4)=1.0                                                          00018520
      B(4,5)=1.0                                                          00018530
      B(5,2)=1.0                                                          00018540
      B(6,5)=1.0                                                          00018550
      RETURN                                                              00018560
      END                                                                 00018570
      SUBROUTINE BOX(KASE,KASE,BC)                                        00018580
C                                                                         00018590
C THIS SUBROUTINE SETS UP THE BLOCK TRIDIAGONAL SYSTEMS OF EQUATIONS FOR00018600
C (A) THE 2-D ATTACHMENT LINE EQUATION WHEN KASE = 2, AND (B) THE 3-D   00018610
C WALL JET PROBLEM WHEN KASE = 3.                                        00018620
C                                                                         00018630
      COMMON /USTORE/ USTORE(6,101,1)                                     00018640
      COMMON /A/ A(6,6,1) /B/ B(4,6,1) /C/ C(2,6,1) /F/ F(6,1)            00018650
      COMMON /MESHY/ H(1) /UT/ U(6,1) /UTA/ UTAN(6,1) /G/ G(6,1)          00018660
      COMMON /SETUP/ UB(6),UBXN(6),UBXXN(6),UBXXXN(6),FF(6),AJA(6,6)      00018670
      COMMON /PARM1/ N,NP,NJ                                              00018680
      COMMON /PARM4/ I,J,IZ                                               00018690
      COMMON /PARM5/ FACX,HKS,XA,XB,YA,YB,ZA,ZB,NX                        00018700
      COMMON /PARM6/ KOUNT,XN,DX,X,ZN,HZ,Z                                00018710
      LOGICAL SWITCH                                                      00018720
      J1=J-1
      DO 30 M=1,J
      DO 30 L=1,M
      DO 10 K=1,NP
      A(K,L,M)=0.
   10 B(K,L,M)=0.
      DO 20 K=1,NQ
      C(K,L,M)=0.
   20 A(K+NP,L,M)=0.
   30 CONTINUE
C
      DO 40 L=1,N
      DO 40 K=1,N
   40 AJA(K,L)=0.
      DO 50 K=1,N
      UH(K)=U(K,1)
   50 UHXN(K)=U(K,J)
C
C BOUNDARY CONDITIONS CONTRIBUTION
C
      CALL BC
      DO 90 K=1,NP
      DO 80 L=1,N
   80 A(K,L,1)=AJA(K,L)
   90 F(K,1)=-FF(K)
      DO 200 K=1,NQ
      DO 100 L=1,N
  100 A(K+NP,L,J)=AJA(K+NP,L)
  200 F(K+NP,J)=-FF(K+NP)
C
      DO 210 L=1,N
      DO 210 K=1,N
  210 AJA(K,L)=0.
C
C INTERNAL POINTS CONTRIBUTION
C
      IZM1=IZ-1
      DO 900 M=2,J
      M1=M-1
      IF(KASE.EQ.2) GO TO 295
      DO 290 K=1,N
      UH(K)=.5*(U(K,M1)+U(K,M))
      UHXN(K)=.5*(UTXN(K,M1)+UTAR(K,M))
      UHXXN(K)=.5*(USTORE(K,M1,IZ)+USTORE(K,M,IZ))
      UHXXM(K)=.5*(USTORE(K,M1,IZM1)+USTORE(K,M,IZM1))
  290 CONTINUE
      GO TO 310
  295 CONTINUE
      DO 300 K=1,N
      UHXN(K)=(UTXN(K,M1)+UTXN(K,M))/2.
  300 UM(K)=(U(K,M))+U(K,M))/2.                                         00019240
  310 CONTINUE                                                          00019250
      CALL RHSF                                                         00019260
      DO 600 K=1,NQ                                                     00019270
      KP=K+NP                                                           00019280
      DO 500 L=1,N                                                      00019290
      HA=-AJA(K,L)*H(M))/2.                                             00019300
      A(KP,L,M))=HA                                                     00019310
  500 C(K,L,M1)=HA                                                      00019320
      F(KP,M1)=U(K,M1)-U(K,M)+H(M1)*FF(K)                               00019330
      A(KP,K,M1)=-1.+A(KP,K,M1)                                         00019340
  600 C(K,K,M1)=1.+C(K,K,M1)                                            00019350
      DO 800 K=1,NP                                                     00019360
      KQ=K+NQ                                                           00019370
      DO 700 L=1,N                                                      00019380
      HA=-AJA(KQ,L)*H(M1)/2.                                            00019390
      A(K,L,M)=HA                                                       00019400
  700 B(K,L,M)=HA                                                       00019410
      F(K,M)=U(KQ,M1)-U(KQ,M)+H(M1)*FF(KQ)                              00019420
      A(K,KQ,M)=1.+A(K,KQ,M)                                            00019430
  800 B(K,KQ,M)=-1.+B(K,KQ,M)                                           00019440
C                                                                       00019450
C   CONTRIBUTIONS FROM PREVIOUS STATIONS                                00019460
C                                                                       00019470
      B(1,3,M)=B(1,3,M)+1.-G(2,M1)                                      00019480
      A(1,3,M)=A(1,3,M)-1.+G(2,M)                                       00019490
      F(1,M)=F(1,M)+U(3,M)-U(3,M1)-G(3,M)+G(2,M1)*U(3,M1)-G(2,M)*U(3,M) 00019500
      B(4,6,M)=B(4,6,M)+1.0-G(6,M1)                                     00019510
      A(4,6,M)=A(4,6,M)-1.0+G(6,M)                                      00019520
      F(4,M)=F(4,M)+U(6,M1)*(G(6,M1)-1.0)-U(6,M)*(G(6,M)-1.0)-G(4,M)    00019530
  900 CONTINUE                                                          00019540
C                                                                       00019550
C REARRANGE EQUATIONS TO ENSURE THE FIRST DIAGONAL BLOCK IS NONSINGULAR 00019560
C THIS IS CRUCIAL FOR THE BLOCK TRIDIAGONAL SOLVER                      00019570
C                                                                       00019580
      KHT=1                                                             00019590
      KQ=NQ+1                                                           00019600
      DO 1500 K=KQ,N                                                    00019610
      SWITCH=.TRUE.                                                     00019620
      DO 1000 L=1,N                                                     00019630
      IF(ABS(A(L,K,1)).GT.1.E-10) SWITCH=.FALSE.                        00019640
      IF(.NOT.SWITCH) GO TO 1500                                        00019650
 1000 CONTINUE                                                          00019660
      KBEGIN=KHT                                                        00019670
      DO 1100 M=KBEGIN,NQ                                               00019680
      DO 1100 L=1,NP                                                    00019690
      IF(ABS(A(L,M,1)).LT.1.E-10) GO TO 1100                            00019700
      KSWTCH=M                                                          00019710
      KHT=M+1                                                           00019720
      GO TO 1200                                                        00019730
 1100 CONTINUE                                                          00019740
 1200 CONTINUE                                                          00019750
      DO 1400 M=1,J1                                                    00019760
      DO 1300 L=1,N                                                     00019770
      T=A(NP+KSWTCH,L,M)                                                00019780
      A(NP+KSWTCH,L,M)=B(K-NQ,L,M+1)                                    00019790
      B(K-NQ,L,M+1)=T                                                   00019800
      T=C(KSWTCH,L,M)                                                   00019810
      C(KSWTCH,L,M)=A(K-NQ,L,M+1)                                       00019820
      A(K-NQ,L,M+1)=T                                                   00019830
 1300 CONTINUE                                                          00019840
      T=F(NP+KSWTCH,M)                                                  00019850
      F(NP+KSWTCH,M)=F(K-NQ,M+1)                                        00019860
      F(K-NQ,M+1)=T                                                     00019870
 1400 CONTINUE                                                          00019880
 1500 CONTINUE                                                          00019890
C                                                                       00019900
      RETURN                                                            00019910
      END                                                               00019920
      SUBROUTINE BLOCK1                                                 00019930
C                                                                       00019940
C THIS SUBROUTINE DECOMPOSES THE BLOCK TRIDIAGONAL MATRIX INTO LU-FORM  00019950
C                                                                       00019960
      COMMON /A/ A(6,6,1) /B/ B(4,6,1) /C/ C(2,6,1)                     00019970
      COMMON /NR/ NR(6,1) /NC/ NC(6,1) /NCR/ NCR(6,1)                   00019980
      COMMON /PARM1/ N,NP,NQ                                            00019990
      COMMON /PARM4/ I,J,IZ                                             00020000
      DO 100 L=1,J                                                      00020010
      DO 100 K=1,N                                                      00020020
      NR(K,L)=K                                                         00020030
      NC(K,L)=K                                                         00020040
  100 NCR(K,L)=0                                                        00020050
      CALL LUSOLV(1)                                                    00020060
C                                                                       00020070
      DO 600 M=2,J                                                      00020080
      M1=M-1                                                            00020090
      KM=M                                                              00020100
C                                                                       00020110
      CALL BETASV(KM)                                                   00020120
C                                                                       00020130
C SOLVE SCALAR MATRIX ALPHA                                             00020140
C                                                                       00020150
      DO 500 K=1,NP                                                     00020160
      DO 400 L=1,N                                                      00020170
      SUM=0.                                                            00020180
      DO 200 KK=1,NQ                                                    00020190
  200 SUM=SUM-B(K,KK+NP,M)*C(KK,L,M1)                                   00020200
      A(K,L,M)=A(K,L,M)+SUM                                             00020210
  400 CONTINUE                                                          00020220
  500 CONTINUE                                                          00020230
C                                                                       00020240
      CALL LUSOLV(KM)                                                   00020250
```

```
C
C
  600 CONTINUE
C
      RETURN
      END
      SUBROUTINE LUSOLV(KM)
C
C THIS SUBROUTINE DECOMPOSES A SCALAR MATRIX INTO LU FORM USING
C A MIXED-PIVOTING STRATEGY
C
      COMMON /A/ A(6,6,1)
      COMMON /NR/ NR(6,1) /NC/ NC(6,1) /NCH/ NCH(6,1)
      COMMON /PARM1/ N,NP,NQ
C
      DO 600 M=2,N
      M1=M-1
C
C SEARCH FOR OPTIMAL PIVOT
C
      KR=M1
      KC=M1
      NRM=NR(M1,KM)
      NCM=NC(M1,KM)
      CPIVOT=A(NRM,NCM,KM)
      RPIVOT=CPIVOT
      DO 200 K=M,N
      NRK=NR(K,KM)
      NCK=NC(K,KM)
      IF(ABS(RPIVOT).GE.ABS(A(NRK,NCM,KM))) GO TO 100
C
      KR=K
      RPIVOT=A(NRK,NCM,KM)
  100 CONTINUE
      IF(ABS(CPIVOT).GE.ABS(A(NRM,NCK,KM))) GO TO 200
      KC=K
      CPIVOT=A(NRM,NCK,KM)
  200 CONTINUE
      IF(ABS(RPIVOT).GE.ABS(CPIVOT)) GO TO 400
C
C PIVOT BY INTERCHANGING COLUMN
C
      IF(ABS(CPIVOT).LT.1.E-10) WRITE(6,6000) KM,M1,CPIVOT
      NCR(M1,KM)=1
      KT=NC(KC,KM)
      IF(KC.NE.M1) KSIGN=KSIGN+1
      NC(KC,KM)=NC(M1,KM)
      NC(M1,KM)=KT
C
C GAUSSIAN ELIMINATION
C
```

```
      DO 300 L=M,N
      NCL=NC(L,KM)
      A(NRM,NCL,KM)=A(NRM,NCL,KM)/CPIVOT
      T=A(NRM,NCL,KM)
      DO 300 K=M,N
      NRK=NR(K,KM)
  300 A(NRK,NCL,KM)=A(NRK,NCL,KM)-T*A(NRK,KT,KM)
      GO TO 600
C
C PIVOT BY INTERCHANGING ROW
C
  400 CONTINUE
      IF(ABS(RPIVOT).LT.1.E-10) WRITE(6,6000) KM,M1,RPIVOT
      KT=NR(KR,KM)
      IF(KR.NE.M1) KSIGN=KSIGN+1
      NR(KR,KM)=NR(M1,KM)
      NR(M1,KM)=KT
C
C GAUSSIAN ELIMINATION
C
      DO 500 L=M,N
      NRL=NR(L,KM)
      A(NRL,NCM,KM)=A(NRL,NCM,KM)/RPIVOT
      T=A(NRL,NCM,KM)
      DO 500 K=M,N
      NCK=NC(K,KM)
  500 A(NRL,NCK,KM)=A(NRL,NCK,KM)-T*A(KT,NCK,KM)
C
  600 CONTINUE
      NRN=NR(N,KM)
      NCN=NC(N,KM)
      IF(ABS(A(NRN,NCN,KM)).LT.1.E-10) WRITE(6,6100) KM,A(NRN,NCN,KM)
 6000 FORMAT(* LUSOLV - BLOCK =*,I4,* PIVOT AT EQN NO.=*,I2,* IS*,E12.5)
 6100 FORMAT(* LUSOLV - BLOCK =*,I4,* LAST PIVOT IS*,E12.5)
C
      RETURN
      END
      SUBROUTINE BETASV(KM)
C
C THIS SUBROUTINE SOLVES FOR BETA IN THE LU-DECOMPOSITION OF THE BLOCK
C TRIDIAGONAL MATRIX
C
      COMMON /SETUP/ UH(6),UD1(6),UD2(6),UD3(6),FF(6),AJA(6,6)
      COMMON /A/ A(6,6,1) /B/ B(4,6,1) /C/ C(2,6,1)
      COMMON /NR/ NR(6,1) /NC/ NC(6,1) /NCR/ NCR(6,1)
      COMMON /PARM1/ N,NP,NU
C
C SOLVE Y IN Y * (U*Q)  =  H
C
      M1=KM-1
      DO 700 M=1,NP
      DO 300 L=1,N
      NCL=NC(L,M1)
      SUM=H(M,NCL,KM)
      IF(L.EQ.1) GO TO 200
      L1=L-1
      DO 100 K=1,L1
      NRK=NR(K,M1)
  100 SUM=SUM-UH(K)*A(NRK,NCL,M1)
  200 UH(L)=SUM
      NRL=NR(L,M1)
      IF(NCR(L,M1).EQ.0) UH(L)=UH(L)/A(NRL,NCL,M1)
  300 CONTINUE
C
C SOLVE BETA IN BETA * A  =  H
C
      DO 500 LL=2,N
      L=N-LL+1
      L1=L+1
      SUM=UH(L)
      NCL=NC(L,M1)
      DO 400 K=L1,N
      NRK=NR(K,M1)
  400 SUM=SUM-UH(K)*A(NRK,NCL,M1)
      UH(L)=SUM
      NRL=NR(L,M1)
      IF(NCR(L,M1).EQ.1) UH(L)=UH(L)/A(NRL,NCL,M1)
  500 CONTINUE
C
C
C REARRANGE COMPONENTS DUE TO MIXED PIVOTING
C
      DO 600 L=1,N
      NRL=NR(L,M1)
  600 B(M,NRL,KM)=UH(L)
  700 CONTINUE
C
      RETURN
      END
      SUBROUTINE BLOCK2
C
C ASSUMING THE BLOCK TRIDIAGONAL MATRIX IS IN FACTORIZED FORM, THIS
C ROUTINE COMPUTES THE SOLUTION FOR A PARTICULAR RIGHT HAND SIDE
      COMMON /B/ B(4,6,1) /C/ C(2,6,1) /F/ F(6,1) /DU/ DU(6,1)
C
      COMMON /PARM1/ N,NP,NQ
      COMMON /PARM4/ I,J,IZ
C
C SOLVE Y IN L * Y  =  F
C
      DO 300 M=2,J
      M1=M-1
      DO 200 K=1,NP                                                      00021790
      SUM=0.                                                             00021800
      DO 100 KK=1,N                                                      00021810
  100 SUM=SUM-H(K,KK,M)*F(KK,M1)                                         00021820
      F(K,M)=F(K,M)+SUM                                                  00021830
  200 CONTINUE                                                           00021840
  300 CONTINUE                                                           00021850
C                                                                        00021860
C SOLVE DU IN A * DU  =  F                                               00021870
C                                                                        00021880
      CALL USOLVE(J)                                                     00021890
C                                                                        00021900
      DO 600 MM=2,J                                                      00021910
C                                                                        00021920
C UPDATE RIGHT HAND SIDE                                                 00021930
C                                                                        00021940
      M1=J-MM+2                                                          00021950
      M=M1-1                                                             00021960
      DO 500 K=1,NM                                                      00021970
      SUM=0.                                                             00021980
      DO 400 L=1,N                                                       00021990
  400 SUM=SUM-C(K,L,M)*DU(L,M1)                                          00022000
  500 F(NP+K,M)=F(NP+K,M)+SUM                                            00022010
C                                                                        00022020
      CALL USOLVE(M)                                                     00022030
C                                                                        00022040
  600 CONTINUE                                                           00022050
C                                                                        00022060
      RETURN                                                             00022070
      END                                                                00022080
      SUBROUTINE USOLVE(M)                                               00022090
C                                                                        00022100
C ASSUMING A SCALAR MATRIX IS IN FACTORIZED FORM, THIS ROUTINE SOLVES    00022110
C THE SOLUTION FOR A PARTICULAR RIGHT HAND SIDE                          00022120
C                                                                        00022130
      COMMON /A/ A(6,6,1) /F/ F(6,1) /DU/ DU(6,1)                        00022140
      COMMON /NR/ NR(6,1) /NC/ NC(6,1) /NCR/ NCR(6,1)                    00022150
      COMMON /PARM1/ N,NP,NM                                             00022160
C                                                                        00022170
C SOLVE Y IN L * Y  =  F                                                 00022180
C                                                                        00022190
      DO 300 L=1,N                                                       00022200
      NRL=NR(L,M)                                                        00022210
      SUM=F(NRL,M)                                                       00022220
      IF(L.EQ.1) GO TO 200                                               00022230
      L1=L-1                                                             00022240
      DO 100 K=1,L1                                                      00022250
      NCK=NC(K,M)                                                        00022260
  100 SUM=SUM-A(NRL,NCK,M)*DU(K,M)                                       00022270
  200 DU(L,M)=SUM                                                        00022280
      NCL NC(L,M)                                                        00022290
      IF(NCR(L,M).EQ.1) DU(L,M)=DU(L,M)/A(NRL,NCL,M)                    00022300
  300 CONTINUE                                                          00022310
      NRN=NR(N,M)                                                       00022320
      NCN=NC(N,M)                                                       00022330
      F(N,M)=DU(N,M)/A(NRN,NCN,M)                                       00022340
C                                                                       00022350
C                                                                       00022360
C SOLVE DU IN A * DU  =  F                                              00022370
C                                                                       00022380
      DO 500 LL=2,N                                                     00022390
      L=N-LL+1                                                          00022400
      L1=L+1                                                            00022410
      SUM=DU(L,M)                                                       00022420
      NRL=NR(L,M)                                                       00022430
      DO 400 K=L1,N                                                     00022440
      NCK=NC(K,M)                                                       00022450
  400 SUM=SUM-A(NRL,NCK,M)*F(K,M)                                       00022460
      F(L,M)=SUM                                                        00022470
      NCL=NC(L,M)                                                       00022480
      IF(NCR(L,M).EQ.0) F(L,M)=F(L,M)/A(NRL,NCL,M)                      00022490
  500 CONTINUE                                                          00022500
C                                                                       00022510
C                                                                       00022520
C REARRANGE COMPONENTS DUE TO MIXED PIVOTING                            00022530
C                                                                       00022540
      DO 600 L=1,N                                                      00022550
      NCL=NC(L,M)                                                       00022560
  600 DU(NCL,M)=F(L,M)                                                  00022570
      RETURN                                                            00022580
      END                                                               00022590
```

INITIAL DISTRIBUTION LIST

Copies	
7	Commander Naval Air Development Center Warminster, PA 18974 (2) Attn: Mr. C. Mazza (Code 6053) (5) Attn: Dr. K. Green (Code 6052)
3	Commander Naval Air Systems Command Jefferson Plaza Washington, DC 20361 Attn: Mr. R. Siewert (AIR-320D)
1	Commander Naval Weapons Center China Lake, CA 93555 Attn: Mr. B. Kowalsky (Code 3183)
1	Commander Naval Ship Research and Development Center Carderock, MD 20034 Attn: Mr. R. Murphy
1	Commander Naval Air Propulsion Test Center Trenton, NJ 98628 Attn: Mr. L. Palcza
1	Chief Office of Naval Research 1800 N. Quincy Street Arlington, VA 22217 Attn: Dr. R. Whitehead
1	Superintendent Naval Postgraduate School Monterey, CA 93940 Attn: Dr. M. Platzer
12	Defense Documentation Center Building #5 Cameron Station Alexandria, VA 22314
1	NASA-Ames Research Center 40 × 80 Wind Tunnel Moffett Field, CA 94035 Attn: Mr. D. Koenig (MS-247-1)

INITIAL DISTRIBUTION LIST (Continued)

Copies

1 Director
Air Force Flight Dynamics Laboratory
AFDL/XFM
Wright-Patterson AFB
Dayton, OH 45433
Attn: Dr. K. Nagaraja

1 Office of Secretary of Defense
Director of Defense Research and Engineering
OAD/ET
Room 3D1089
Pentagon, Washington, DC 20301
Attn: Dr. B. Osborne, Jr.

3 Science Center
Rockwell International Corporation
1049 Camino Dos Rios
P.O. Box 1085
Thousand Oaks, CA 91360
Attn: Dr. N. Malmuth

1 Vought Corporation
Advanced Technology Center, Inc.
P.O. Box 6144
Dallas, TX
Attn: Dr. C. Haight

1 General Dynamics Corporation
Fort Worth, TX 76101
Attn: Mr. J. Clifton (MS-2892)

1 Rockwell International
Columbus, OH 43216
Attn: Dr. P. Bevilaqua

1 Flight Dynamics Research Corporation
15809 Stagg Street
Van Nuys, CA 91406
Attn: Dr. M. Alperin

1 Computational Mechanics
3601A Chapman Highway
Knoxville, TN 37920
Attn: Dr. J. Baker